HITACHI
Inspire the Next

Hitachi Application Server V10
ユーザーズガイド（UNIX®用）

3021-3-415-30

前書き

■ 対象製品

適用 OS：Red Hat Enterprise Linux 5 Advanced Platform（AMD/Intel 64）、Red Hat Enterprise Linux 5（AMD/Intel 64）、Red Hat Enterprise Linux Server 6（64-bit x86_64）、Red Hat Enterprise Linux Server 7（64-bit x86_64）

P-9W43-5KA1 Hitachi Application Server 10-11

注　このマニュアルで「AIX の場合」と表記している個所については、この製品ではサポートしていません。

適用 OS については、上記以外にもご利用になれる場合があります。詳細は『リリースノート』でご確認ください。

■ 輸出時の注意

本製品を輸出される場合には、外国為替及び外国貿易法の規制並びに米国輸出管理規則など外国の輸出関連法規をご確認の上、必要な手続きをお取りください。

なお、不明な場合は、弊社担当営業にお問い合わせください。

■ 商標類

Microsoft および SQL Server は、米国 Microsoft Corporation 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。

Microsoft および Visual C++は、米国 Microsoft Corporation 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。

Oracle と Java は、Oracle Corporation 及びその子会社、関連会社の米国及びその他の国における登録商標です。

RC4 は、米国 EMC コーポレーション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。

Red Hat は、米国およびその他の国で Red Hat, Inc. の登録商標もしくは商標です。

RSA および BSAFE は、米国 EMC コーポレーション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。

UNIX は、The Open Group の米国ならびに他の国における登録商標です。

W3C は、World Wide Web Consortium の商標（多数の国において登録された）です。

Windows は、米国 Microsoft Corporation 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。

Windows Server は、米国 Microsoft Corporation 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。

その他記載の会社名、製品名などは、それぞれの会社の商標もしくは登録商標です。

本製品は、米国 EMC コーポレーションの RSA BSAFE®ソフトウェアを搭載しています。

This product includes software developed by Ben Laurie for use in the Apache-SSL HTTP server project.

Portions of this software were developed at the National Center for Supercomputing Applications (NCSA) at the University of Illinois at Urbana-Champaign.

This product includes software developed by the University of California, Berkeley and its contributors.

This software contains code derived from the RSA Data Security Inc. MD5 Message-Digest Algorithm, including various modifications by Spyglass Inc., Carnegie Mellon University, and Bell Communications Research, Inc (Bellcore).

Regular expression support is provided by the PCRE library package, which is open source software, written by Philip Hazel, and copyright by the University of Cambridge, England. The original software is available from ftp://ftp.csx.cam.ac.uk/pub/software/programming/pcre/

This product includes software developed by Ralf S. Engelschall <rse@engelschall.com> for use in the mod_ssl project (http://www.modssl.org/).

This product includes software developed by IAIK of Graz University of Technology.

This product includes software developed by Daisuke Okajima and Kohsuke Kawaguchi (http://relaxngcc.sf.net/).

This product includes software developed by Andy Clark.

Java is a registered trademark of Oracle and/or its affiliates.

■ 発行

2017 年 9 月 3021-3-415-30

変更内容

変更内容(3021-3-415-30)

Hitachi Application Server 10-11

追加・変更内容	変更個所
リリースノートのマニュアル訂正を反映した。	－

単なる誤字・脱字などはお断りなく訂正しました。

はじめに

■ マニュアルの目的

このマニュアルは Hitachi Application Server の機能概要、およびシステム開発・運用工程に沿ったユースケースをベースとして各工程にマッピングした手順について説明します。このマニュアルを利用することで、機能概要や製品アーキテクチャを理解して、インストールから運用までのひととおりの操作ができるようになることを目的としています。

■ 対象読者

このマニュアルは次の方を対象読者としています。

- システム構築者

前提知識を次に示します。

- システム構築者
  - Windows または UNIX の操作に関する知識
  - Application Server の構築に関する知識
  - Java EE 標準仕様に関する知識
  - システム構築で使用する周辺環境（データベース、ネットワーク、ジョブ管理など）に関する知識

■ 関連マニュアルの表記

関連マニュアル、およびこのマニュアルで使用している関連マニュアル名の表記を次の表に示します。

Hitachi Application Server 関連

表記	正式名称	資料番号
『ユーザーズガイド』	『Hitachi Application Server V10 ユーザーズガイド』（UNIX®用）	3021-3-415
『GUI リファレンス』	『Hitachi Application Server V10 GUI リファレンス』（UNIX®用）	3021-3-417
『コマンドリファレンス』	『Hitachi Application Server V10 コマンドリファレンス』（UNIX®用）	3021-3-419
『定義リファレンス』	『Hitachi Application Server V10 定義リファレンス』（UNIX®用）	3021-3-421
『メッセージリファレンス』	『Hitachi Application Server V10 メッセージリファレンス』	3021-3-422
『API リファレンス』	『Hitachi Application Server V10 API リファレンス』	3021-3-423

HiRDB 関連

表記	正式名称	資料番号
『HiRDB システム導入・設計ガイド』	『ノンストップデータベース HiRDB Version 9 システム導入・設計ガイド』（UNIX®用）	3000-6-452
	『ノンストップデータベース HiRDB Version 9 システム導入・設計ガイド』（Windows®用）	3020-6-452
『HiRDB メッセージ』	『ノンストップデータベース HiRDB Version 9 メッセージ』	3020-6-458

JP1 関連

表記	正式名称	資料番号
『JP1/Automatic Job Management System 導入ガイド』	『JP1 Version 10 JP1/Automatic Job Management System 3 導入ガイド』	3021-3-102
『JP1/Automatic Job Management System 設計ガイド』	『JP1 Version 10 JP1/Automatic Job Management System 3 設計ガイド（システム構築編）』	3021-3-103
	『JP1 Version 10 JP1/Automatic Job Management System 3 設計ガイド（業務設計編）』	3021-3-104
『JP1/Automatic Job Management System 構築ガイド 1』	『JP1 Version 10 JP1/Automatic Job Management System 3 構築ガイド 1』	3021-3-105
『JP1/Automatic Job Management System 操作ガイド』	『JP1 Version 10 JP1/Automatic Job Management System 3 操作ガイド』	3021-3-109

■ 製品名と機能名の表記

このマニュアルでは、製品名と機能名を次のように表記しています。

表記	製品名と機能名
Application Server	Hitachi Application Server
Application Server - Base	Hitachi Application Server - Base
Application Server - Optional License for Java	Hitachi Application Server - Optional License for Java
Application Server for Developers	Hitachi Application Server for Developers
APV	IBM Advanced POWER Virtualization
DAS	Domain Administration Server
ドメイン管理サーバ	

表記			製品名と機能名
Developer's Kit for Java			Hitachi Developer's Kit for Java
Excel			Microsoft® Excel
Firefox			Firefox®
HiRDB	HiRDB Version 9		HiRDB Server Version 9
	HiRDB/Single Server		HiRDB/Single Server Version 9
HWS			Hitachi Web Server
Web Server			
Internet Explorer			Windows® Internet Explorer®
Java EE Server			Hitachi Java EE Server
JP1/AJS3			JP1 Version 10 JP1/Automatic Job Management System 3 - Agent
			JP1 Version 10 JP1/Automatic Job Management System 3 - Manager
			JP1 Version 10 JP1/Automatic Job Management System 3 - View
JP1/IM			JP1 Version 10 JP1/Integrated Management - Manager
			JP1 Version 10 JP1/Integrated Management - View
Microsoft IIS	Microsoft IIS 7.5		Microsoft® Internet Information Services 7.5
	Microsoft IIS 8.0		Microsoft® Internet Information Services 8.0
	Microsoft IIS 8.5		Microsoft® Internet Information Services 8.5
Microsoft Visual C++			Microsoft® Visual C++®
Oracle	Oracle 11g		Oracle Database 11g
			Oracle Database 11g R2
	Oracle 12c		Oracle Database 12c
SQL Server			Microsoft® SQL Server
UNIX	AIX		AIX V6.1
			AIX V7.1
	Linux	Linux（AMD64 & Intel EM64T）	Red Hat Enterprise Linux® 5 Advanced Platform（AMD/Intel 64）
			Red Hat Enterprise Linux® 5（AMD/Intel 64）
			Red Hat Enterprise Linux® Server 6（64-bit x86_64）

表記			製品名と機能名
			Red Hat Enterprise Linux® Server 7（64-bit x86_64）
Virtage			Hitachi Virtage
VMware ESX			VMware vSphere® ESX
VMware vSphere ESXi			VMware vSphere® ESXi™
Windows	Windows Server 2008 R2		Microsoft® Windows Server® 2008 R2 Standard 日本語版
			Microsoft® Windows Server® 2008 R2 Enterprise 日本語版
			Microsoft® Windows Server® 2008 R2 Datacenter 日本語版
	Windows Server 2012		Microsoft® Windows Server® 2012 Standard 日本語版
			Microsoft® Windows Server® 2012 Datacenter 日本語版
	Windows Server 2012 R2		Microsoft® Windows Server® 2012 R2 Standard 日本語版
			Microsoft® Windows Server® 2012 R2 Datacenter 日本語版
	Windows 7	Windows 7 x86	Microsoft® Windows® 7 Professional 日本語版(32 ビット版)
			Microsoft® Windows® 7 Enterprise 日本語版(32 ビット版)
			Microsoft® Windows® 7 Ultimate 日本語版(32 ビット版)
		Windows 7 x64	Microsoft® Windows® 7 Professional 日本語版(64 ビット版)
			Microsoft® Windows® 7 Enterprise 日本語版(64 ビット版)
			Microsoft® Windows® 7 Ultimate 日本語版(64 ビット版)
	Windows 8	Windows 8 x86	Windows® 8 Pro 日本語版(32 ビット版)
			Windows® 8 Enterprise 日本語版(32 ビット版)
		Windows 8 x64	Windows® 8 Pro 日本語版(64 ビット版)
			Windows® 8 Enterprise 日本語版(64 ビット版)
		Windows 8.1 x86	Windows® 8.1 Pro 日本語版(32 ビット版)
			Windows® 8.1 Enterprise 日本語版(32 ビット版)
		Windows 8.1 x64	Windows® 8.1 Pro 日本語版(64 ビット版)
			Windows® 8.1 Enterprise 日本語版(64 ビット版)
	Windows 10	Windows 10 x64	Windows® 10 Pro 日本語版(64 ビット版)
			Windows® 10 Enterprise 日本語版(64 ビット版)
Windows Server Failover Cluster			Windows Server® Failover Cluster
クラス別統計			日立クラス別統計

表記	製品名と機能名
パフォーマンストレーサ	Performance Tracer
パフォーマンストレーサー	

Linux に関しては、バージョンごとに次のように表記することがあります。

表記		OS 名
Red Hat Enterprise Linux 5	Red Hat Enterprise Linux 5 Advanced Platform（AMD/Intel 64）	Red Hat Enterprise Linux® 5 Advanced Platform（AMD/Intel 64）
	Red Hat Enterprise Linux 5（AMD/Intel 64）	Red Hat Enterprise Linux® 5（AMD/Intel 64）
Red Hat Enterprise Linux Server 6	Red Hat Enterprise Linux Server 6（64-bit x86_64）	Red Hat Enterprise Linux® Server 6（64-bit x86_64）
Red Hat Enterprise Linux Server 7	Red Hat Enterprise Linux Server 7（64-bit x86_64）	Red Hat Enterprise Linux® Server 7（64-bit x86_64）

■ 英略語

このマニュアルで使用している英略語を次に示します。

英略語	英字での表記
ACC	Application Client Container
ACL	Access Control List
AES	Advanced Encryption Standard
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
BLOB	Binary Large Object
CA	Certificate Authority
CDI	Contexts and Dependency Injection
CGI	Common Gateway Interface
CMP	Container-Managed Persistence
CMT	Container-Managed Transaction
CopyGC	Copy Garbage Collection
CORBA	Common Object Request Broker Architecture
CPU	Central Processing Unit

英略語	英字での表記
CRL	Certificate Revocation List
CSR	Certificate Signing Request
CSV	Comma Separated Value
CVS	Concurrent Versions System
DBMS	Database Management System
DCOM	Distributed Component Object Model
DD	Deployment Descriptor
DDE	Dynamic Data Exchange
DER	Distinguished Encoding Rules
DES	Data Encryption Standard
DI	Dependency Injection
DLL	Dynamic Link Library
DMZ	Demilitarized Zone
DN	Distinguished Name
DNS	Domain Name System
DoS	Denial of Service
DSO	Dynamic Shared Object
DTD	Document Type Definition
EAR	Enterprise Archive
ear	
EIS	Enterprise Information System
EJB	Enterprise JavaBeans™
EJB QL	EJB™ Query Language
EL	Expression Language
EUC	Extended UNIX Code
FQDN	Fully Qualified Domain Name
FullGC	Full Garbage Collection
G1GC	Garbage First Garbage Collection
GC	Garbage Collection

英略語	英字での表記
GMS	Group Management Service
GMT	Greenwich Mean Time
GUI	Graphical User Interface
HA	High Availability
HTML	Hyper Text Markup Language
HTTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Security
I/O	Input/Output
ID	Identifier
IDE	Integrated Development Environment
IEC	International Electrotechnical Commission
IIOP	Internet Inter-Orb Protocol
IIS	Internet Information Services
IMAP	Internet Message Access Protocol
IP	Internet Protocol
IPv6	Internet Protocol Version 6
ISO	International Organization for Standardization
J2EE	J2EE™
	Java™ 2 Platform, Enterprise Edition
JAAS	Java™ Authentication and Authorization Service
JACC	Java™ Authorization Service Provider Contract for Containers
JAF	JavaBeans™ Activation Framework Specification
JAR	Java™ Archive
jar	
JASPIC	Java™ Authentication Service Provider Interface for Containers
Java	Java™
Java EE	Java™ Platform, Enterprise Edition
Java EE RI	Java EE Reference Implementation
Java HotSpot Client VM	Java HotSpot™ Client Virtual Machine

英略語	英字での表記
Java Platform Debugger Architecture	Java™ Platform Debugger Architecture
JPDA	
Java SE	Java™ Platform, Standard Edition
Java VM	Java™ Virtual Machine
JVM	
JavaMail	JavaMail™
JAX-RPC	Java™ API for XML-based RPC
JAX-RS	Java™ API for RESTful Web Services
JAX-WS	Java™ API for XML-based Web Services
JAXB	Java™ Architecture for XML Binding
JAXP	Java™ API for XML Processing
JAXR	Java™ API for XML Registries
JCA	J2EE™ Connector Architecture
JDBC	Java™ Database Connectivity
	JDBC™
JDK	Java™ Development Kit
	JDK™
JIS	Japanese Industrial Standards
JMS	Java™ Message Service
JMX	Java™ Management Extensions
JNDI	Java Naming and Directory Interface™
JNI	Java™ Native Interface
JPA	Java™ Persistence API
JSF	JavaServer™ Faces
	JavaServer™ Faces Reference Implementation (RI) Version: 1.1_01 FCS
JSON-P	Java™ API for JSON Processing
JSP	JavaServer Pages™
	JSP™

英略語	英字での表記
JST	Japan Standard Time
JSTL	JavaServer Pages™ Standard Tag Library
JTA	Java™ Transaction API
JVMPI	Java™ Virtual Machine Profiler Interface
JVMTI	Java™ Virtual Machine Tool Interface
KVM	Kernel-based Virtual Machine
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
MAC	Message Authentication Code
MIME	Multipurpose Internet Mail Extensions
MPM	Multi-Proccessing Module
OASIS	Organization for the Advancement of Structured Information Standards
OMG	Object Management Group
ORB	Object Request Broker
OS	Operating System
OTS	Object Transaction Service
QNAME	Qualified Name
REST	Representational State Transfer
RMI	Remote Method Invocation
RPC	Remote Procedure Call
RSA	Rivest, Shamir and Adleman
SAAJ	SOAP with Attachments API for Java™
SAX	Simple API for XML
SEI	Service Endpoint Interface
Servlet	Java™ Servlet
SHA	Secure Hash Algorithm
SMAP	Source Map
SMTP	Simple Mail Transfer Protocol
SOAP	Simple Object Access Protocol

英略語	英字での表記
SSH	Secure Shell
ssh	
SSL	Secure Sockets Layer
StAX	Streaming API for XML
TCP	Transmission Control Protocol
TLD	Tag Library Descriptor
TLS	Transport Layer Security
UCS	Universal multi-octet coded Character Set
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
UTC	Coordinated Universal Time
UTF	UCS Transformation Format
UTF-8	8-bit UCS Transformation Format
VM	Virtual Machine
VTL	Velocity Template Language
W3C	World Wide Web Consortium
WAR	Web Archive
war	
WBEM	Web-Based Enterprise Management
WSDL	Web Services Description Language
XML	Extensible Markup Language

■ KB（キロバイト）などの単位表記について

1KB（キロバイト）、1MB（メガバイト）、1GB（ギガバイト）、1TB（テラバイト）、1PB（ペタバイト）はそれぞれ 1,024 バイト、$1{,}024^2$ バイト、$1{,}024^3$ バイト、$1{,}024^4$ バイト、$1{,}024^5$ バイトです。

目次

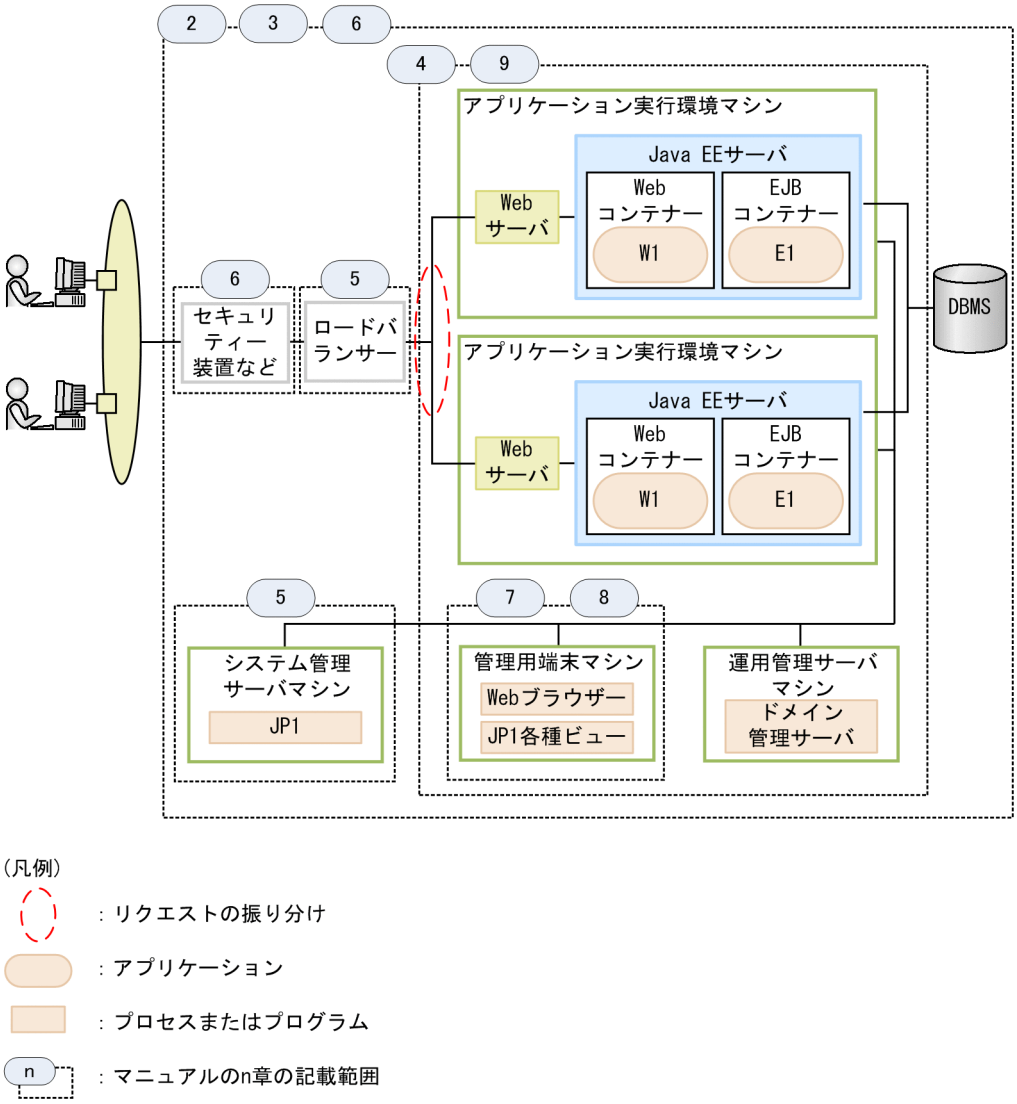
1 マニュアルの読み方

Application Server を利用したシステムの開発から運用までの工程に沿って、このマニュアルのどこを読めばよいかを説明します。

1.1　マニュアルの読み方について

Application Server を利用した Web フロントシステムを開発する際に、このマニュアルで参照する章について説明します。Web フロントシステムの開発から運用までの工程ごとに、このマニュアルでの参照先（章）を示します。また、各章の概要についても説明します。なお、アプリケーションを実行する環境をアプリケーション実行環境と呼びます。

このマニュアルで説明する、Application Server を利用した Web フロントシステムの構成を次の図に示します。



（凡例）

：リクエストの振り分け

：アプリケーション

：プロセスまたはプログラム

n　：マニュアルのn章の記載範囲

アプリケーション実行環境の設計・構築・運用時に参照する個所

アプリケーション実行環境の設計・構築・運用の工程ごとに実施する内容と、参照先の章の概要を次の表に示します。利用目的に合わせて章を選択して読むことができます。なお、システムの構築および運用で実施する作業は、コマンドまたは Administration Console のどちらも利用できます。

工程	実施する内容	図中の番号	マニュアルの章タイトル	対応するマニュアルの記載概要
システムの設計	システム構成の検討、およびシステムの設計を実施します。	2	Application Server の概要	システムの構成を検討するために必要な情報について説明します。 Application Server の製品情報や、目的別のシステム構成例、プロセス構成などの情報を基に、システムの構成を検討します。
		3	Application Server の設計項目	システムを設計するために必要な情報について説明します。 システムの信頼性や性能を確保するために、必要に応じて設計項目を選択してシステムの設計をします。
システムの構築	アプリケーションを実行する環境を構築し、使用する機能を設定します。	4	アプリケーション実行環境の構築	アプリケーションの実行環境を構築する手順について説明します。 1 つの Java EE サーバを配置するシステム、または複数の Java EE サーバを配置して負荷を分散させるクラスター構成のシステムを構築します。なお、クラスター構成とは、複数の Java EE サーバをグループ化して管理するための構成のことです。
		5	システム周辺環境の設定	システム周辺環境で設定する項目について説明します。ネットワーク、負荷分散、および運用の自動化を設定します。
		6	高信頼性機能の設定	高信頼性を実現するための、障害検知に関する機能（プロセス監視、メッセージ監視）、およびセキュリティーを強化する設定（リバースプロキシー、SSL）について説明します。
システムの運用	通常運用時、または保守運用時の作業を実施します。	7	通常運用時の作業	運用開始後に日々実施する、通常運用時の作業について説明します。 Application Server に対して、次の作業を実施してシステムを運用します。 • システムの起動・停止 • マシンの起動・停止と同期したシステムの起動・停止 • Application Server の稼働状況の確認 • アプリケーションの稼働状況の確認 • データベースの接続状況の確認
		8	保守運用時の作業	稼働状況の変化やシステム構成の変更に対応するための保守運用時の作業について説明します。 Application Server に対して、次の作業を実施してシステムをメンテナンスします。 • 稼働状況やシステム構成に応じた環境定義の変更

工程	実施する内容	図中の番号	マニュアルの章タイトル	対応するマニュアルの記載概要
				• IP アドレス・ホスト名の変更 • アプリケーションの入れ替え • 環境情報のバックアップ・リストア • 修正パッチ・修正版の適用 • システムの利用状況や動作状況の確認 • システムのスケールアウト • バージョンアップ
	出力されるトラブルシュート資料を調査します。	9	トラブルシュート資料の活用	アプリケーションの実行環境および開発環境で出力される、各種ログ、トレースなどのトラブルシュート資料について説明します。出力されるトラブルシュート資料の見方やパフォーマンストレーサーのトレース取得ポイントなどの情報を参考にして、トラブルシュート資料を調査します。

重要

このマニュアルでは、1 つのメインパスを想定して、Web フロントシステムの設計、構築、および運用を説明しています。Web フロントシステム以外のシステムや、システム設計時に設計するすべての設計項目、システム構築に関するすべての手順や設定値、およびシステム運用で実施するすべての運用手順を説明しているわけではありません。

2 Application Server の概要

Application Server を利用したシステムの構成を検討するために必要な情報について説明します。 Application Server の製品情報や、目的別のシステム構成例、プロセス構成などの情報を基に、システムの構成を検討します。

2.1 Application Server が対応する標準仕様

Application Server は、Java EE 7、Java SE、Web Service、CORBA、およびインターネット関連の標準仕様に対応しています。Application Server が対応する標準仕様について説明します。

Java EE 7 仕様

項番	標準仕様
1	Java EE 7
2	WebSocket 1.0※1
3	JSON-P 1.0
4	Servlet 3.1
5	JSF 2.2
6	EL 3.0
7	JSP 2.3
8	JSTL 1.2
9	Batch 1.0※2
10	Concurrency Utilities 1.0
11	CDI 1.1
12	DI 1.0
13	Bean Validation 1.1
14	Interceptors 1.2
15	JCA 1.7
16	JPA 2.1※3
17	Common Annotations for the Java Platform 1.2
18	JMS 2.0
19	JTA 1.2
20	JavaMail 1.5
21	JAX-RS 2.0
22	Web Services for Java EE 1.4
23	JAX-WS 2.2
24	Web Services Metadata for the Java Platform 2.1
25	JASPIC 1.1
26	JACC 1.5

項番	標準仕様
27	Java EE Management 1.1
28	Debugging Support for Other Languages 1.0
29	Managed Beans 1.0
30	EJB 3.2※4
31	JAX-RPC 1.1
32	JAXR 1.0
33	Java EE Deployment 1.2

注※1

HTTP サーバとして Microsoft IIS 8.0 以降を使用する場合に対応しています。

注※2

データベースとして HiRDB を使用する場合に対応しています。

注※3

データベースとして HiRDB を使用する場合、Criteria API とストアドプロシージャーの呼び出しには対応していません。

注※4

CMP および EJB QL には対応していません。

Java SE 仕様

- Java SE 8

Java EE7 仕様に関連する Java SE 仕様

項番	標準仕様
1	JAXB 2.2
2	JAXP 1.3
3	JMX 1.2
4	JAF 1.1
5	StAX 1.0
6	SAAJ 1.3
7	JDBC 4.1

Web Service 系仕様

項番	標準仕様	
1	SOAP（W3C 標準）	SOAP 1.1
2		SOAP 1.2
3	WSDL（W3C 標準）	WSDL 1.1
4	Bootstrapping	WS-MetadataExchange v1.1
5	Policy（W3C 標準）	WS-Policy v1.2
6		WS-PolicyAttachment v1.2
7		WS-Policy v1.5
8		WS-PolicyAttachment v1.5
9	Reliable Messaging （OASIS 標準）	WS-ReliableMessaging v1.0
10		WS-ReliableMessaging Policy v1.0
11		WS-ReliableMessaging v1.1
12		WS-ReliableMessaging Policy v1.1
13		WS-ReliableMessaging v1.2
14		WS-ReliableMessaging Policy v1.2
15		WS-MakeConnection v1.1
16	Atomic Transactions（OASIS 標準）※	WS-AtomicTransaction v1.0
17		WS-Coordination v1.0
18	Security（OASIS 標準 ）	WS-Security v1.0
19		WS-Security v1.1
20		WS-SecurityPolicy v1.1
21		WS-SecurityPolicy v1.2
22		WS-Trust v1.2
23		WS-Trust v1.3
24		WS-Trust v1.4
25		WS-SecureConversation v1.2
26		WS-SecureConversation v1.3
27		WS-SecureConversation v1.4
28	Security Profiles（OASIS 標準）	Web Services Security: SOAP Message Security V1.0
29		WS-Security Core Specification 1.1

項番	標準仕様	
30		Username Token Profile V1.0
31		Username Token Profile 1.1
32		X.509 Token Profile V1.0
33		X.509 Token Profile 1.1
34		SAML Token Profile V1.0
35		SAML Token profile 1.1
36		Kerberos Token Profile 1.1
37		Rights Expression Language (REL) Token Profile 1.1
38		SOAP with Attachments (SWA) Profile 1.1
39	Addressing（W3C 標準 ）	Web Services Addressing 1.0 - Core
40		Web Services Addressing 1.0 - SOAP Binding
41		Web Services Addressing 1.0 - WSDL Binding
42	WS-I Profile	WS-I Basic Profile 1.2
43		WS-I Basic Profile 2.0
44		WS-I Attachments Profile 1.0
45		WS-I Simple SOAP Binding Profile 1.0

※

Metro がサポートしているのは、標準化前の項番 16、17 のバージョンです。

CORBA 系仕様

- Common Object Request Broker Architecture (CORBA), Version 3.0

インターネット関連の標準仕様

項番	標準仕様
1	IPv6
2	HTTP 1.0
3	HTTP 1.1
4	SSL v3
5	TLS 1.0
6	TLS 1.1
7	TLS 1.2

2.2 Application Server の製品構成、前提プログラム、関連製品について

Application Server の製品構成や前提 OS について説明します。また、使用できる仮想化プラットフォーム、DBMS、Web ブラウザー、HTTP サーバ、メールサーバ、関連製品などの、Application Server を使用する際に前提となる製品情報についても説明します。

製品構成

Application Server は、次の製品で構成されます。

Application Server

アプリケーションを実行するための環境（アプリケーション実行環境）を構築する製品です。

Application Server for Developers

Application Server で実行するアプリケーションを開発するための環境（アプリケーション開発環境）を構築する製品です。この製品は Windows 版だけです。

前提 OS

Application Server は、次の OS で利用できます。

Application Server の前提 OS

OS のバージョン
Red Hat Enterprise Linux 5 Red Hat Enterprise Linux Server 6 Red Hat Enterprise Linux Server 7
AIX 6.1 AIX 7.1

対応文字コードセット

Application Server は、次の文字コードセット で利用できます。

OS	文字コードセット
Linux	UTF-8
AIX	• Shift-JIS • UTF-8 • EUC

仮想化プラットフォーム

Application Server の仮想化プラットフォームには、次のハイパーバイザーを利用できます。

ハイパーバイザー	対応するゲスト OS
Virtage	Linux
KVM	Linux
VMware ESX • VMware ESX 4.0 • VMware ESX 4.1 • VMware vSphere ESXi 5	Linux
IBM Advanced POWER Virtualization (APV)	AIX

DBMS

Application Server は、次の DBMS と接続できます。

DBMS	JDBC ドライバー
HiRDB Version 9	HiRDB Type4 JDBC Driver
Oracle 11g	Oracle JDBC Drivers 12.1.0.1.0 以降
Oracle 12c	Oracle JDBC Drivers 12.1.0.1.0 以降 または Oracle JDBC Drivers 12.1.0.1.0 以降

Web ブラウザー

Application Server は、次の Web ブラウザーから構築および運用の操作が実行できます。

なお、Administration Console は UNIX では使用できません。

Web ブラウザー	必要なプログラム
Internet Explorer 7 以降	Adobe Flash Player 11.8 以降
Firefox 34.0 以降 Firefox ESR 31 以降	

HTTP サーバ

Application Server は、次の HTTP サーバと接続できます。

HTTP サーバ	バージョン
Web Server	Application Server に同梱される HTTP サーバ

メールサーバ

Application Server は、次のメールサーバを利用できます。

メールサーバ	目的
SMTP サーバ、IMAP サーバ	JavaMail の使用

関連製品

Application Server は、次の製品と関連してシステムを運用できます。

製品名	目的
JP1/IM	システム全体の障害の監視
JP1/AJS3	システムの運用の自動化
HA モニタ	クラスター構成のシステムの運用

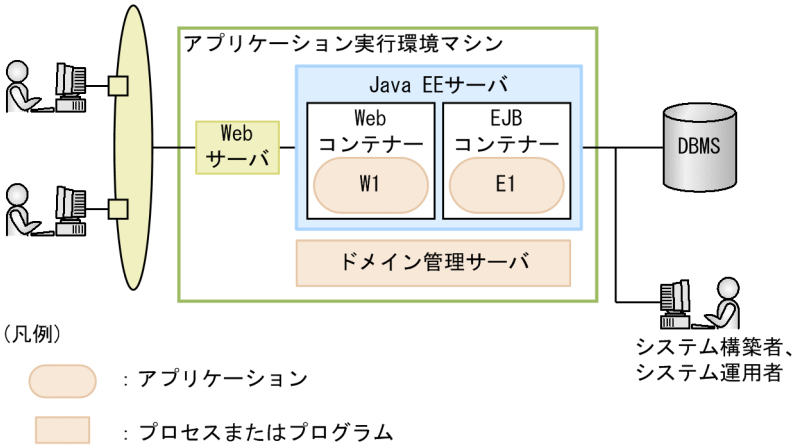
2.3　Application Server のシステム構成

Application Server を利用すると、目的とする業務や実行するアプリケーションの要件に応じた構成でシステムを構築できます。Application Server と Web サーバを同一マシンに配置する最小構成のほか、可用性および性能を確保するための構成、セキュリティーを確保するための構成、安定稼働のための構成、他システムと連携するための構成、マルチテナント構成などを構築できます。

2.3.1　最小のシステム構成について

Java EE サーバと Web サーバを同一マシンに配置する、最小の構成例について説明します。

Java EE サーバと Web サーバを同一マシンに配置する、最小の構成です。トランザクション数が少ないシステムに適しています。

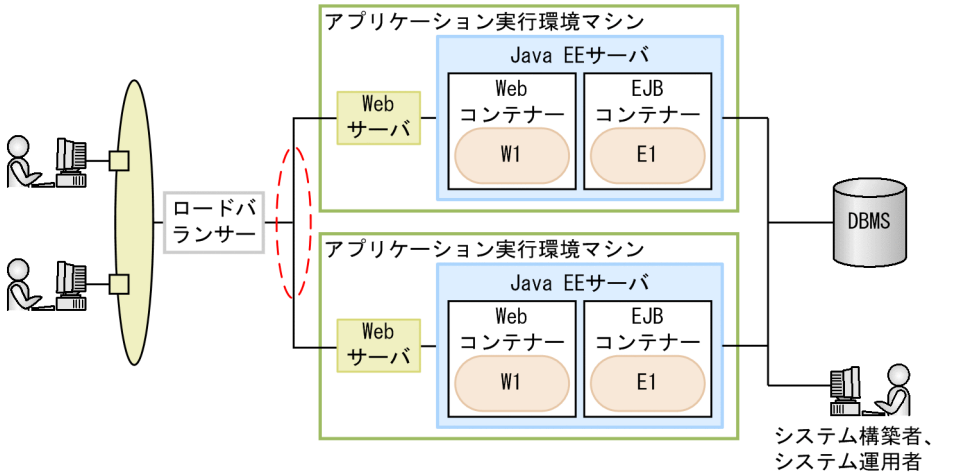


2.3.2　可用性および性能を確保するためのシステム構成について

Application Server を利用した、可用性および性能を確保するためのシステム構成について説明します。代表的なシステム構成の例として、ハードウェアロードバランサーを使用した構成、任意のソフトウェアロードバランサーを使用した構成、Web Server のロードバランス機能を使用した構成、およびクラスターソフトウェアを使用した構成とそれぞれの特徴について説明します。

ハードウェアロードバランサーを使用した構成

ハードウェアロードバランサーを使用して、複数の Java EE サーバにリクエストを振り分ける構成です。トランザクション数が多く、ミッションクリティカルなシステムに適しています。高い信頼性および可用性を確保できます。



注　ドメイン管理サーバは、アプリケーション実行環境マシン内に配置するか、または別マシンに配置できます。

（凡例）

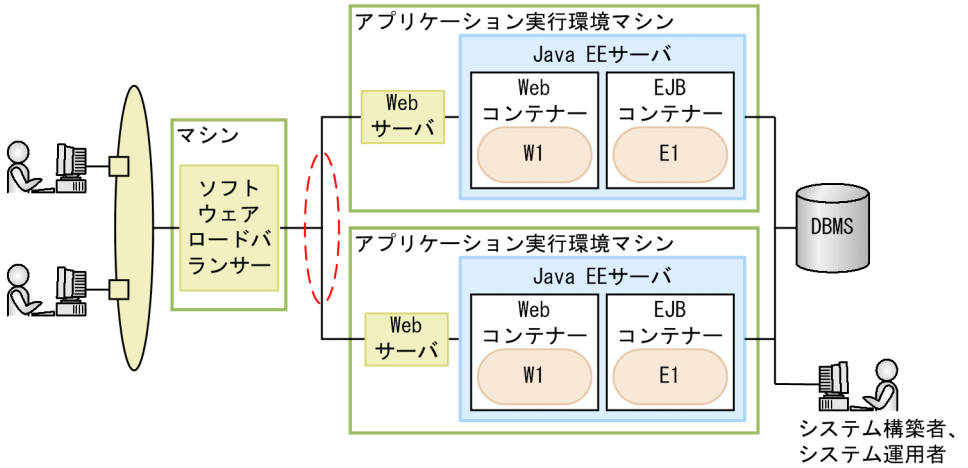
：リクエストの振り分け

：アプリケーション

：プロセスまたはプログラム

任意のソフトウェアロードバランサーを使用した構成

任意のソフトウェアロードバランサーを使用して、複数の Java EE サーバにリクエストを振り分ける構成です。トランザクション数が多いシステムに適しています。高価なロードバランサーを使用しないで、信頼性および可用性を確保します。



注　ドメイン管理サーバは、アプリケーション実行環境マシン内に配置するか、または別マシンに配置できます。

（凡例）

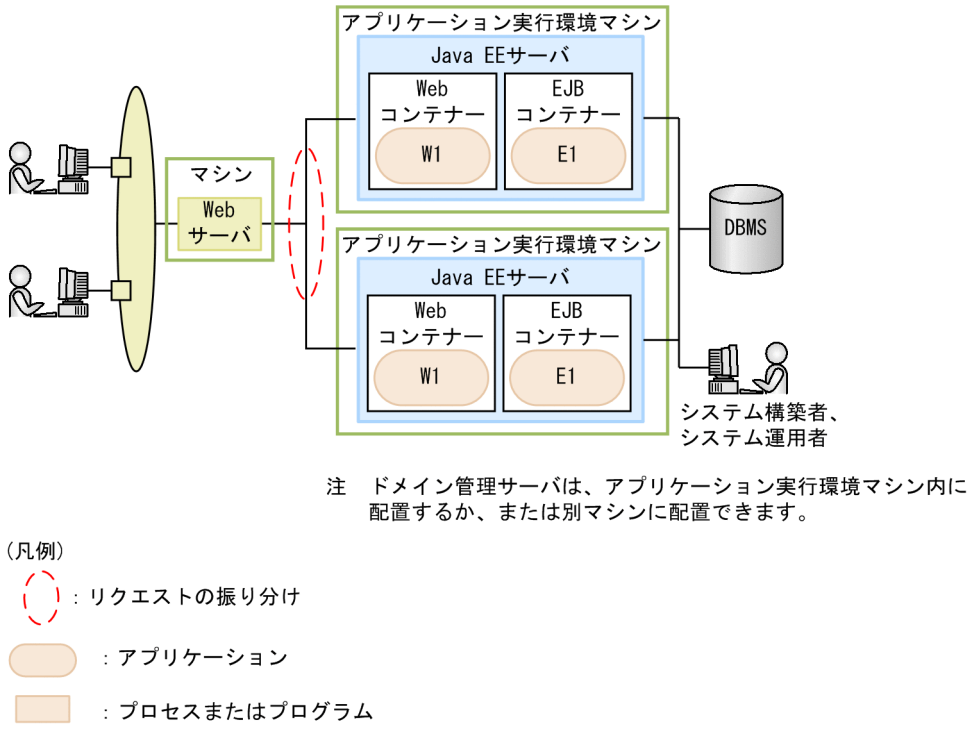
：リクエストの振り分け

：アプリケーション

：プロセスまたはプログラム

Web Server のロードバランス機能を使用した構成

Web Server のロードバランス機能を使用して、複数の Java EE サーバにリクエストを振り分ける構成です。トランザクション数が多いシステムに適しています。Web Server の機能を使用することで、安価で、信頼性および可用性を確保します。



注　ドメイン管理サーバは、アプリケーション実行環境マシン内に配置するか、または別マシンに配置できます。

（凡例）

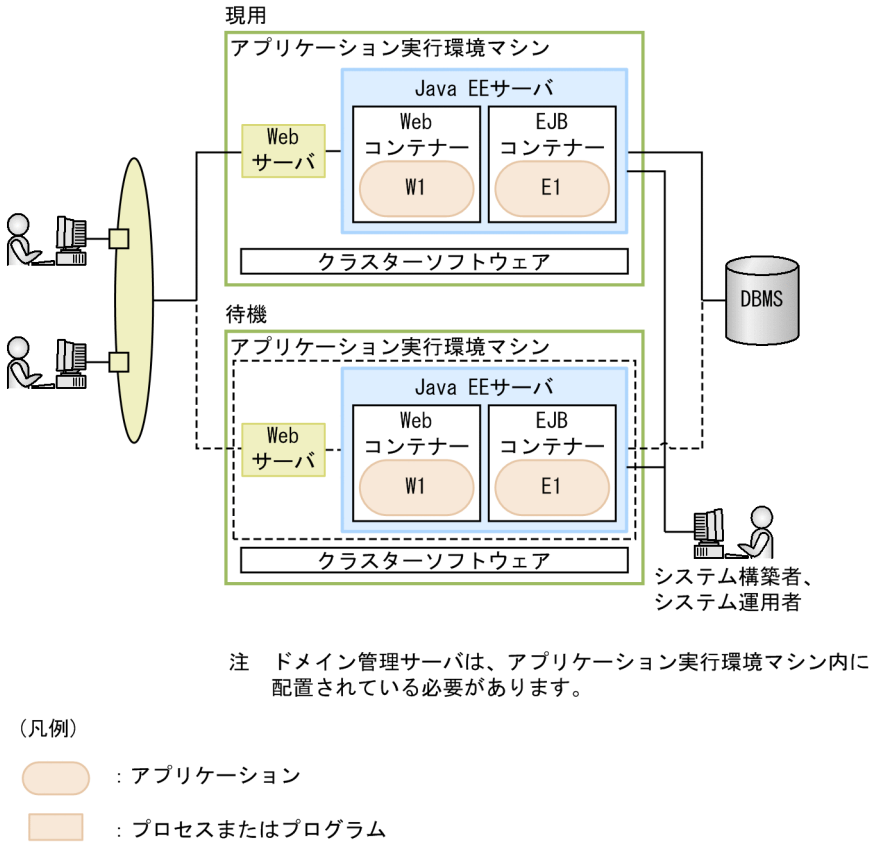
：リクエストの振り分け

：アプリケーション

：プロセスまたはプログラム

クラスターソフトウェアを使用した構成

クラスターソフトウェアを使用して、複数の Java EE サーバを、現用（Active）/待機（Standby）構成に配置した構成です。トランザクション数が少なく、マシン 1 台の性能でリクエストを処理できるシステムに適しています。また、障害発生時に備えて予備系（待機系）を準備し、通常は開発やテスト環境として有効活用できます。



注　ドメイン管理サーバは、アプリケーション実行環境マシン内に配置されている必要があります。

（凡例）

：アプリケーション

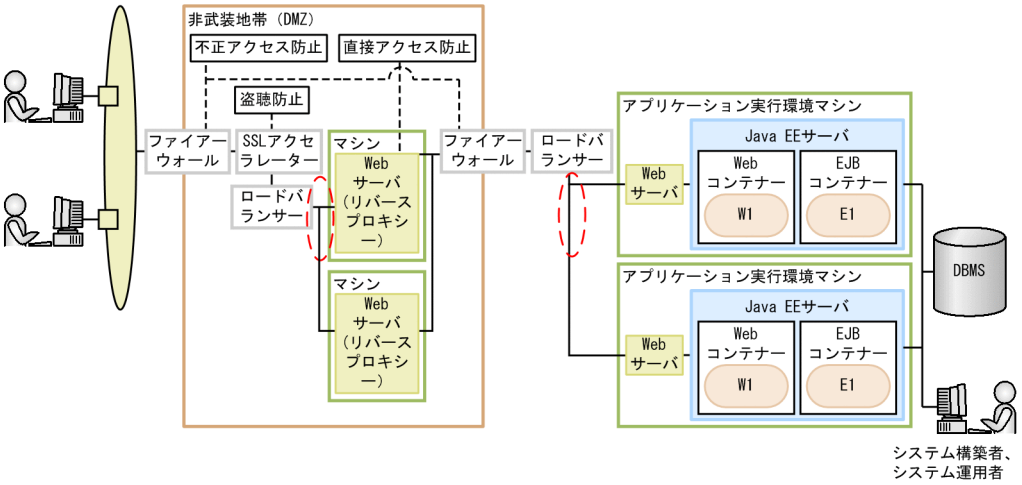
：プロセスまたはプログラム

2.3.3　セキュリティーを確保するための構成について

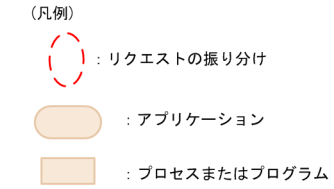
Application Server を利用した、セキュリティーを確保するためのシステム構成について説明します。代表的なシステム構成の例として、リバースプロキシーを使用した構成、およびリバースプロキシーを使用しない構成とそれぞれの特徴について説明します。

リバースプロキシーを使用した構成

DMZ 内にリバースプロキシー、内部ネットワークに Web サーバと Java EE サーバを配置してセキュリティーを確保する構成です。不正アクセス、盗聴、DoS 攻撃を防止します。また、複数の Java EE サーバを配置して可用性を確保します。

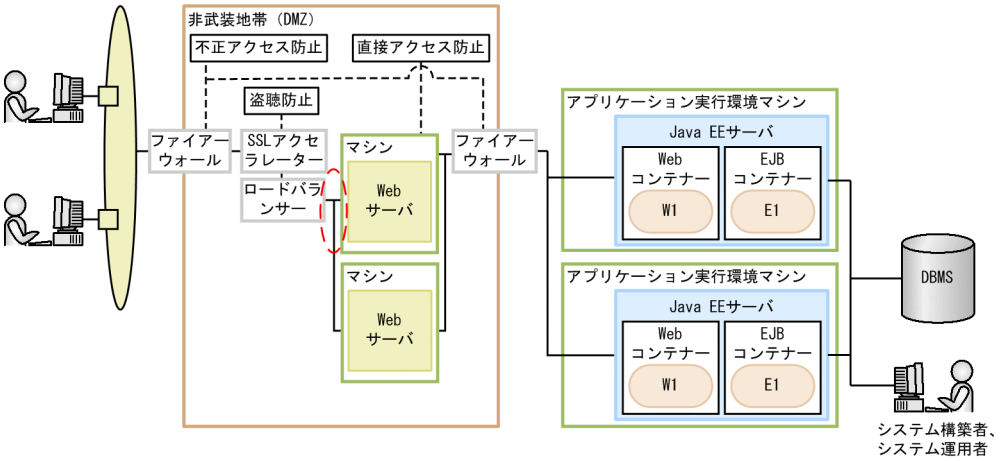


注　ドメイン管理サーバは、アプリケーション実行環境マシン内に配置するか、または別マシンに配置できます。

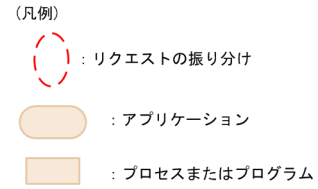


リバースプロキシーを使用しない構成

DMZ 内に Web サーバ、内部ネットワークに Java EE サーバを配置してセキュリティーを確保する構成です。通信のオーバーヘッドを削減するため、リバースプロキシーは配置しません。不正アクセス、盗聴、DoS 攻撃を防止します。また、複数の Java EE サーバを配置して可用性を確保します。



注　ドメイン管理サーバは、アプリケーション実行環境マシン内に配置するか、または別マシンに配置できます。

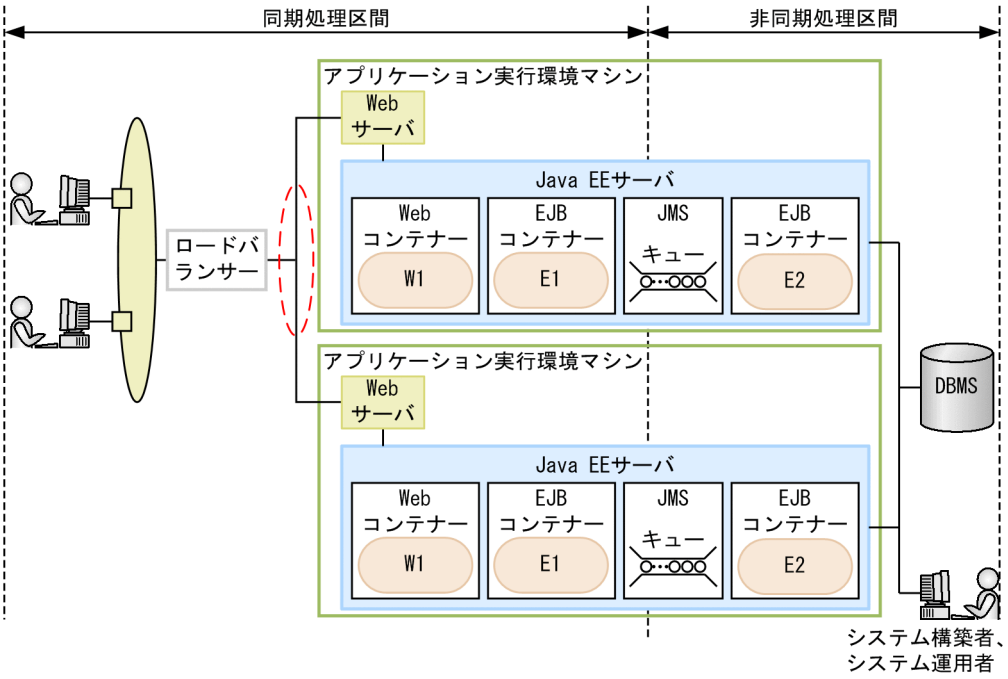


2.3.4　安定稼働のための構成について

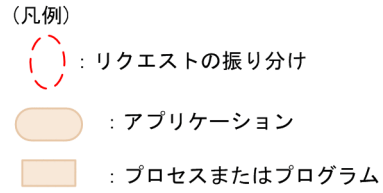
Application Server を利用した、安定稼働のためのシステム構成について説明します。代表的なシステム構成の例として、非同期でオンラインバッチ処理を行う構成とその特徴について説明します。

非同期でオンラインバッチ処理を行う構成

負荷の高い処理を JMS のキューにいったん蓄積して、同時実行数を絞った状態で非同期処理をする構成です。短期間に送信された大量リクエストや、長時間掛る処理などによって発生する同時実行数の増大を防ぎ、安定稼働ができます。



注　ドメイン管理サーバは、アプリケーション実行環境マシン内に配置するか、または別マシンに配置できます。

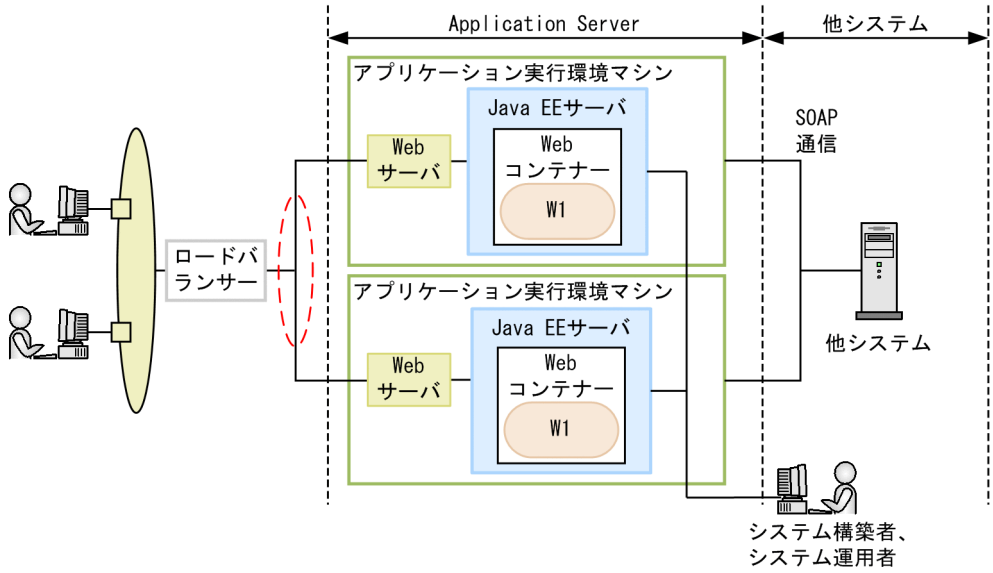


2.3.5　他システムと連携するための構成について

Application Server を利用した、他システムと連携するためのシステム構成について説明します。代表的なシステム構成の例として、バックに他システムと接続する構成、フロントに他システムと接続する構成、および WebSphere MQ を使用して他システムと接続する構成とそれぞれの特徴について説明します。

バックに他システムと接続する構成

フロントにある Application Server でリクエストを受け付け、そのバックにある業務システムを呼び出す構成です。SOAP、CORBA、REST といったインターフェースを利用するため、既存システムと疎結合で連携することができます。



注　ドメイン管理サーバは、アプリケーション実行環境マシン内に配置するか、または別マシンに配置できます。

（凡例）

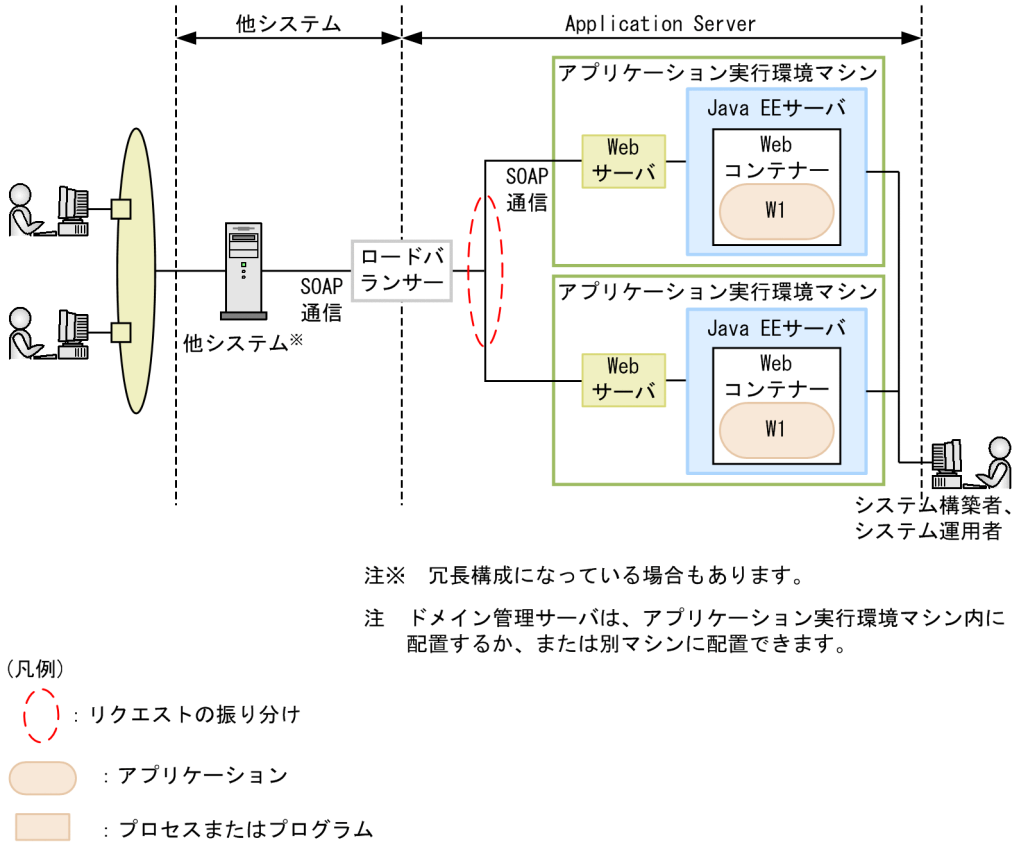
：リクエストの振り分け

：アプリケーション

：プロセスまたはプログラム

フロントに他システムと接続する構成

既存システムからのリクエストを受け、バックの Application Server で業務を実行する構成です。SOAP、REST といったインターフェースを利用するため、既存システムと疎結合で連携することができます。



注※　冗長構成になっている場合もあります。

注　ドメイン管理サーバは、アプリケーション実行環境マシン内に配置するか、または別マシンに配置できます。

（凡例）

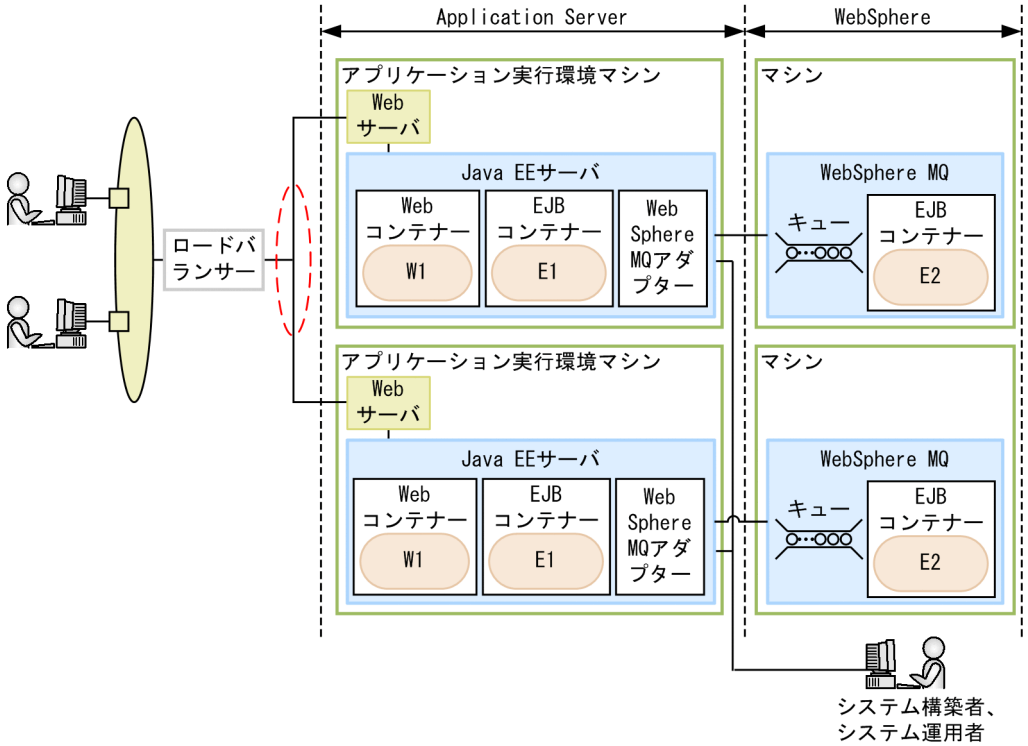
：リクエストの振り分け

：アプリケーション

：プロセスまたはプログラム

WebSphere MQ を使用して他システムと接続する構成

リソースアダプターである WebSphere MQ アダプターを Application Server 上にデプロイし、WebSphere MQ と接続する構成です。WebSphere MQ インターフェースを持った既存システムと連携できます。



注　ドメイン管理サーバは、アプリケーション実行環境マシン内に配置するか、または別マシンに配置できます。

（凡例）

：リクエストの振り分け

：アプリケーション

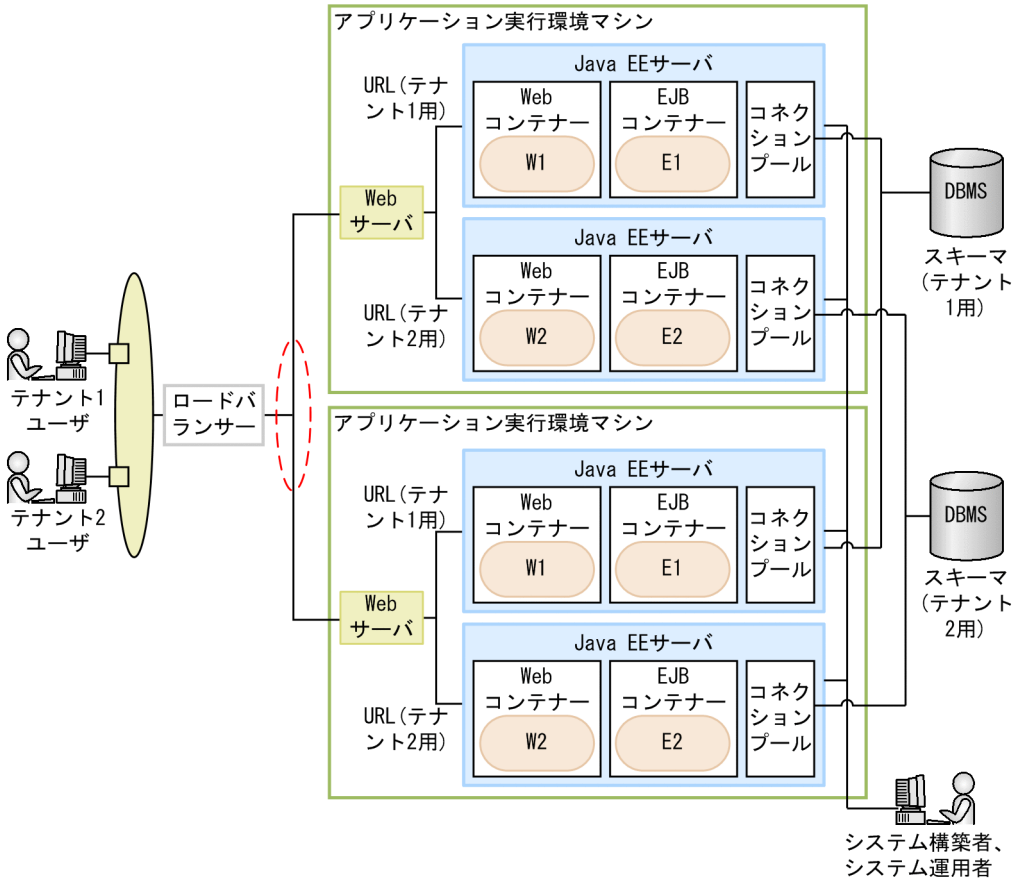
：プロセスまたはプログラム

2.3.6　マルチテナント構成について

Application Server を利用した、マルチテナント構成について説明します。代表的なシステム構成の例として、テナントごとに Java EE サーバを分ける構成とその特徴について説明します。

テナントごとに Java EE サーバを分ける構成

Web サーバのバックの Java EE サーバをテナントごとに分ける構成です。それぞれのテナントのユーザーは異なる URL でシステムにアクセスします。Web サーバは、URL に応じてそれぞれの Java EE プロセスにリクエストを振り分けます。Java EE サーバを分けることによって、プロセスダウンなどの障害発生時にほかのテナントへの影響を抑止できます。



注　ドメイン管理サーバは、アプリケーション実行環境マシン内に配置するか、または別マシンに配置できます。

（凡例）

：リクエストの振り分け

：アプリケーション

：プロセスまたはプログラム

2.4 Application Server の管理要素とプロセス構成について

Application Server を利用したシステムは、ドメイン、ノード、サーバ、クラスター、構成、およびサーバ間関連という要素で管理します。ドメインの管理には、ドメイン管理サーバを利用します。また、Application Server が動作するためのプロセスには、Web サーバ、Java EE サーバ、パフォーマンストレーサー、ドメイン管理サーバなどがあります。

Application Server の管理要素

Application Server を利用したシステムでは、システム内の構成要素をドメインという単位で管理します。ドメインを管理するためには、ドメイン内に 1 つあるドメイン管理サーバを利用します。ドメイン管理サーバが管理するドメインは、ノードおよびサーバという管理要素で構成されます。また、クラスター、サーバ間関連、および構成という要素で、サーバを管理します。

ドメイン

業務システムを構成する複数のサーバを管理する要素です。ドメインは、異なるマシン上のサーバ（Java EE サーバ、Web サーバ、パフォーマンストレーサー）も管理できます。ドメインの管理には、ドメイン管理サーバを利用します。

ドメイン管理サーバ

ドメイン管理サーバとは、ドメインを管理するために、特別に用意されたサーバインスタンスです。ドメイン管理サーバは 1 つのドメインに対して 1 つ存在し、server という名称で作成されます。なお、ドメイン管理サーバは、アプリケーション実行環境マシンとは別の運用管理サーバマシン上に構築することを推奨します。

ノード

ドメイン内の接続先のホスト（マシン）を定義する要素です。サーバはノードで定義されたホストに構築されます。ドメイン内に複数のノードを定義できます。複数のドメインから 1 つのノードを共有することはできません。

サーバ

ドメインが管理する各種サーバのインスタンス要素です。この要素はサーバの実体を表し、サーバの設定情報を包含します。サーバ要素には、Java EE サーバ、Web サーバおよびパフォーマンストレーサーがあります。サーバは必ず 1 つのドメインに属します。

サーバの種類	概要
Java EE サーバ	Java EE アプリケーションを稼働させるコンテナーサーバです。
Web サーバ	Web コンテンツを提供するサービスプログラムです。Web Server を使用します。
パフォーマンストレーサー	Java EE サーバ、Web サーバなどのサーバの稼働性能を監視するプログラムです。監視対象となるサーバと同じノードに配置します。

クラスター

同じアプリケーション、リソース、および設定情報を共有する Java EE サーバのグループ要素です。異なるマシン上の Java EE サーバをグループ化して管理できます。クラスターを利用して、複数の Java EE サーバをグループ化したシステムの構成をクラスター構成と呼びます。クラスターは、スケーラビリティー、負荷分散、およびフェイルオーバーの保護を目的に使用します。

サーバ間関連

2 つのサーバ間の関連で必要な設定情報をカプセル化したインスタンス要素です。構築したサーバ同士を連携して動作させるために必要な設定として、関連先のサーバの情報を関連元のサーバに自動的に設定します。例えば、関連先サーバの待ち受けポート番号を、接続ポートとして関連元のサーバにパラメーターで自動設定します。サーバ間関連には、リダイレクト関連とパフォーマンストレーサー関連があります。

関連の種類	概要
リダイレクト関連	Web サーバが受信したリクエストのリダイレクト先 Java EE サーバを設定するための関連です。異なるノード間のサーバ同士で関連を設定できます。 関連元：Web サーバ 関連先： Java EE サーバまたはクラスター
パフォーマンストレーサー関連	運用中のサーバの稼働性能を確認するために、パフォーマンストレーサーで性能解析トレースを取得できるように設定する関連です。異なるノード間のサーバ同士で関連を設定できません。 関連元：Web サーバまたは Java EE サーバ 関連先：パフォーマンストレーサー

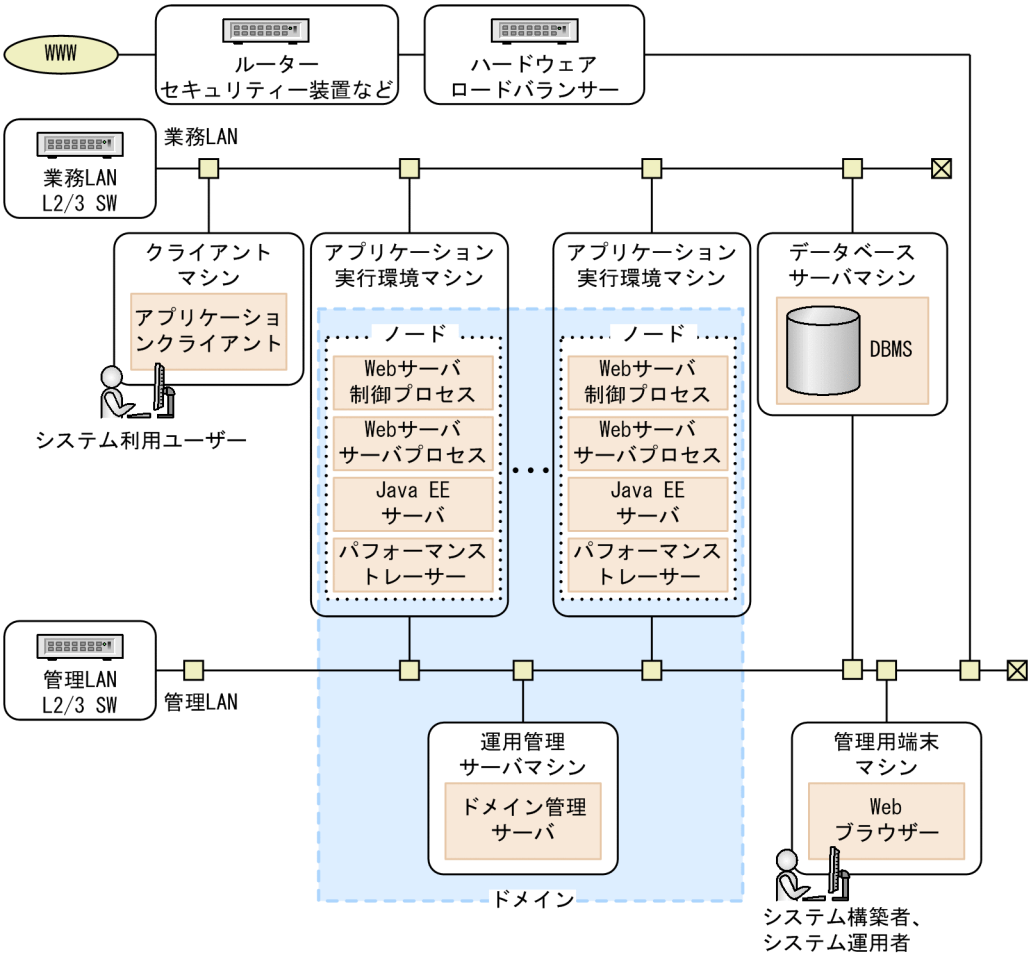
構成

各種サーバの設定情報をカプセル化したインスタンス要素です。サーバやクラスターは、この要素を参照して設定情報を取得します。ただし、構成と関連づいたサーバに、構成と同じパラメーターで異なる値が設定されていた場合は、サーバに設定されたパラメーターが有効になります。構成には、Java EE サーバ構成、Web サーバ構成、およびパフォーマンストレーサー構成があります。

構成の種類	概要
Java EE サーバ構成	Java EE サーバまたはクラスターと関連づく構成で、設定情報を包含しています。Java EE サーバまたはクラスターを作成すると自動的にそのサーバまたはクラスター用の構成が作成されます。 クラスターに Java EE サーバを登録した場合は、その Java EE サーバはクラスターに関連付いている構成を引き継ぎます。
Web サーバ構成	Web サーバと関連づく構成で、設定情報を包含しています。Web サーバを作成すると自動的にそのサーバ用の構成が作成されます。
パフォーマンストレーサー構成	パフォーマンストレーサーと関連づく構成で、設定情報を包含しています。パフォーマンストレーサーを作成すると自動的にそのサーバ用の構成が作成されます。

Application Server のプロセス構成

Application Server のプロセス構成の例を次の図に示します。



root 権限で運用する場合の Application Server のプロセスの一覧を次の表に示します。

構成要素	プロセス名			プロセスの権限	役割
アプリケーションクライアント	java			一般ユーザー権限	アプリケーションクライアントを実行します。
Web サーバ制御プロセス	httpsd			管理者権限	リクエストを処理するサーバプロセスの起動やその稼働監視をします。
Web サーバサーバプロセス	---	httpsd		User ディレクティブに指定したユーザーの権限	Web ブラウザーからの静的コンテンツ要求に対する応答や、アプリケーション要求に対する Java EE サーバへの転送をします。 httpsd は、設定および要求リクエスト数によってプロセス数が変わります。
	---	rotatelogs		管理者権限	
	---	rotatelogs2		管理者権限	
	---	---	gcache	User ディレクティブに指定したユーザーの権限	

構成要素	プロセス名			プロセスの権限	役割
					rotatelogs および rotatelogs2 は、 設定によってプロセス数が変わります。rotatelogs のデフォルトは 2 個です。rotatelogs2 のデフォルトは 1 個です。
Java EE サーバ	java	---	---	管理者権限	アプリケーションを実行します。
パフォーマンストレーサー	cprfd	---	---	管理者権限	Web サーバおよび Java EE サーバの一連の実行履歴を取得します。
ドメイン管理サーバ	java	---	---	管理者権限	Web サーバ、Java EE サーバ、およびパフォーマンストレーサーの運用管理をします。

Java EE サーバのプロセスのカレントディレクトリーを次の表に示します。

Java EE サーバのプロセス	起動時のカレントディレクトリー
サーバインスタンス※	*Java EEサーバのインストールディレクトリー*/glassfish/nodes/*ノード名*/*サーバインスタンス名*/config
ドメイン管理サーバ※	*Java EEサーバのインストールディレクトリー*/glassfish/domains/*ドメイン名*/config
MQ Broker	*Java EEサーバのインストールディレクトリー*/glassfish/nodes/*ノード名*/*サーバインスタンス名*/config

注※

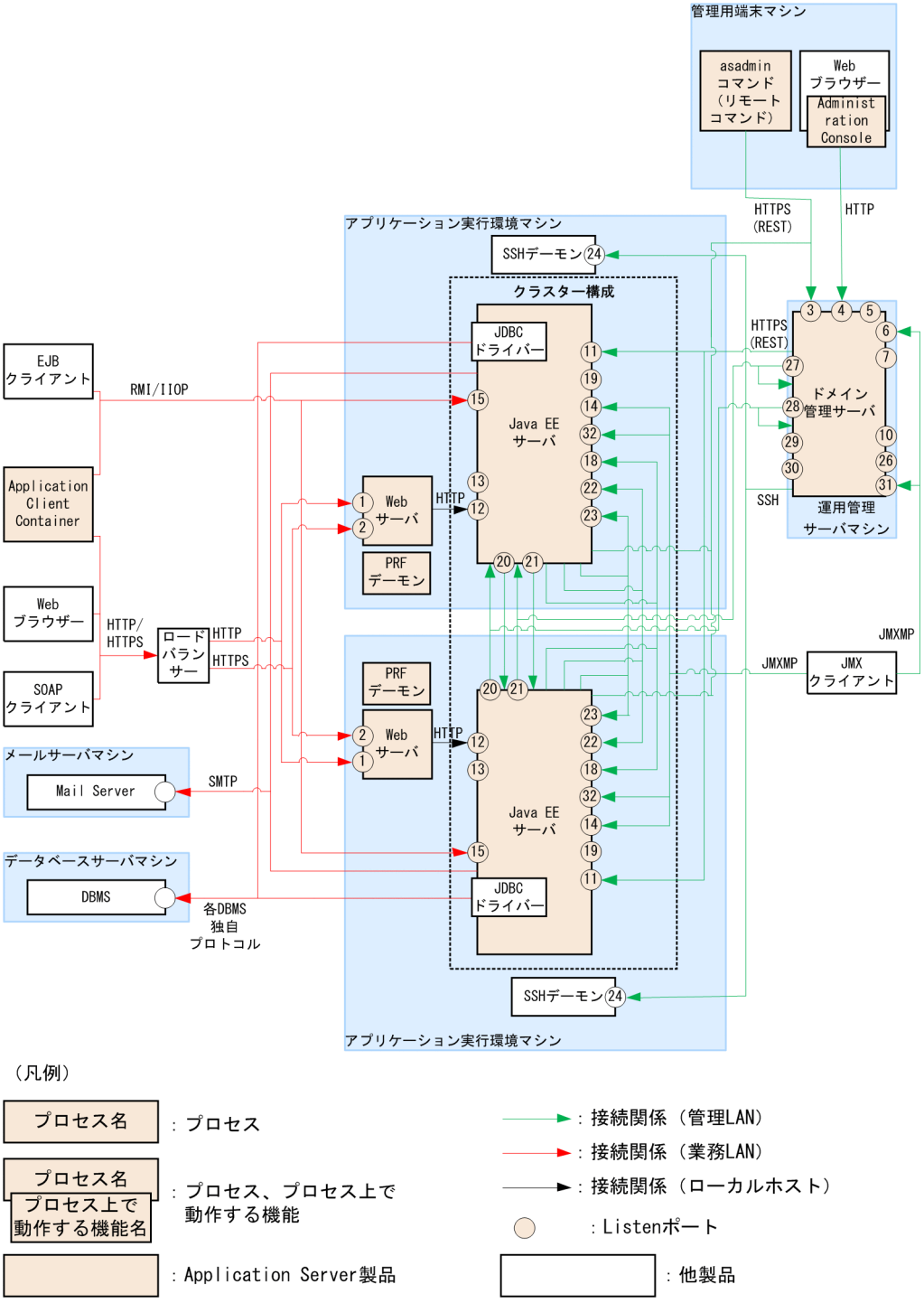
プロセスの起動時、および再起動時には、カレントディレクトリーに出力されているファイルのうち次のログファイルを除いて、すべて削除されます。ただし、次のログファイルのファイル名を変更した場合は、削除の対象となります。また、asadmin ユーティリティーコマンドのstart-instance サブコマンドに、--sync=full オプションを指定して起動すると、カレントディレクトリーのファイルはログファイルも含めて削除されます。

項番	ログ
1	スレッドダンプログ
2	メモリーダンプログ
3	Java VM 出力メッセージログ（エラーリポートファイル）
4	コンパイラーリプレイファイル

2.5 Application Server の接続構成について

Application Server は、Application Server を構成する Java EE サーバ、Web サーバ、ドメイン管理サーバなどのプロセスや、ロードバランサー、DBMS、クライアントなどの周辺機器の間で通信を行って動作しています。これらのプロセス間の接続関係と使用するポート番号について説明します。

Application Server の接続関係を次の図に示します。



Application Server の各プロセスが使用するポートのポート番号、ポート番号の変更可否、バインドする IP アドレスの指定可否、ポートの開閉可否、および通信のクライアントについて、次の表に示します。

プロセス	図中の番号	説明	デフォルトのポート番号/TCP または UDP	ポート番号の変更可否	IP アドレスの指定可否	ポートの開閉可否※1	デフォルト開閉※2	通信のクライアント
Web サーバ	1	HTTP のリクエスト受け付けポート	80/TCP	○	○	○	○	自マシンまたは他マシンの Web クライアント
	2	HTTPS のリクエスト受け付けポート	443/TCP	○	○	○	×	自マシンまたは他マシンの Web クライアント
ドメイン管理サーバ	3	ドメイン管理サーバ用ポート	4848/TCP	○	○	×※3	○	アプリケーション実行環境マシンを含む自マシンまたは他マシンの asadmin ユーティリティーコマンド
	4	HTTP のリクエスト受け付けポート	8080/TCP	○	○	○	○	自マシンまたは他マシンの Administration Console
	5	HTTPS のリクエスト受け付けポート	8181/TCP	○	○	○	×	なし
	6	JMX のリクエスト受け付けポート	8686/TCP	○	○	○	○	自マシンまたは他マシンの JMX クライアント
	7	IIOP 用通信ポート	3700/TCP	○	○	○	○	なし
	10	Java デバッガ用通信ポート	9009/TCP	○	○	○	×	なし
	26	JMS Provider ポート	7676/TCP	○	○	○	○	なし
	27	GMS listener ポート	9090～9200 の間/TCP	○	○	○	×	自マシンまたは他マシンの Java EE サーバ

プロセス	図中の番号	説明	デフォルトのポート番号/TCP または UDP	ポート番号の変更可否	IP アドレスの指定可否	ポートの開閉可否※1	デフォルト開閉※2	通信のクライアント
	28	GMS multicast ポート	2048～49151 の間/UDP	○	○	○	×	自マシンまたは他マシンの Java EE サーバ
	29	Message Queue TCP ポート	動的短命ポートを使用/TCP	○	○	○※4	×	なし
	30	Message Queue Admin ポート	動的短命ポートを使用/TCP	○	○	○※4	×	なし
	31	Message Queue RMI Registry ポート	動的短命ポートを使用/TCP	×	×	○※4	×	自マシンまたは他マシンの JMX クライアント
Java EE サーバ	11	ドメイン管理サーバ用ポート	24848※5/TCP	○	○	×※3	○	自マシンまたは他マシンのドメイン管理サーバ
	12	HTTP のリクエスト受け付けポート	28080※5/TCP	○	○	○	○	自マシンまたは他マシンの Web Server
	13	HTTPS のリクエスト受け付けポート	28181※5/TCP	○	○	○	×	なし
	14	JMX 用通信ポート	28686※5/TCP	○	○	○	○	自マシンまたは他マシンの JMX クライアント
	15	IIOP 用通信ポート	23700※5/TCP	○	○	○	×	• 自マシンまたは他マシンの EJB クライアント • 自マシンまたは他マシンの Application Client Container

プロセス	図中の番号	説明	デフォルトのポート番号/TCP または UDP	ポート番号の変更可否	IP アドレスの指定可否	ポートの開閉可否※1	デフォルト開閉※2	通信のクライアント
	18	JMS Provider ポート	27676※5/TCP	○	○	○	○	自マシンまたは他マシンの Java EE サーバ
	19	Java デバッガ用通信ポート	29009/TCP	○	×※6	○	×	自マシンまたは他マシンの Java Remote Debugger
	20	GMS listener ポート	9090～9200 の間/TCP	○	○	○	×	自マシンまたは他マシンの Java EE サーバ
	21	GMS multicast ポート	2048～49151 の間/UDP	○	○	○	×	自マシンまたは他マシンの Java EE サーバ
	22	Message Queue TCP ポート	動的短命ポートを使用/TCP	○	○	○※4	○	自マシンまたは他マシンの Java EE サーバ
	23	Message Queue Admin ポート	動的短命ポートを使用/TCP	○	○	○※4	○	自マシンまたは他マシンの Java EE サーバ
	32	Message Queue RMI Registry ポート	動的短命ポートを使用/TCP	×	×	○※4	○	自マシンまたは他マシンの JMX クライアント
SSH デーモン	24	SSH サーバの待ち受けポート	ユーザー指定 (22)/TCP	※7	※7	※7	※7	自マシンまたは他マシンのドメイン管理サーバ

(凡例)

○：ポート番号の変更、IP アドレスの指定、またはポートの開閉ができます。

×：ポート番号の変更、IP アドレスの指定、またはポートの開閉ができません。

注※1

ユーザーがポートを開閉できるかどうか示します。

注※2

○の場合、デフォルトで開いています。×の場合、デフォルトで閉じています。

注※3

このポートは該当するプロセスで必須のため、プロセスを起動すると必ず開きます。

注※4

JMS Provider ポートの開閉と連動します。

注※5

コマンドのオプションで指定できます。指定を省略した場合、1 つ目の Java EE サーバはデフォルト値で作成されます。2 つ目以降の Java EE サーバは、サーバ作成時に、ドメイン管理サーバによって自動で設定されます。

注※6

このポートはデフォルトで閉じています。

注※7

SSH デーモンの実装によって異なります。

2.6　インストール後のディレクトリー構成

Application Server をインストールした直後のディレクトリー構成を示します。

Application Server のディレクトリー構成

ディレクトリー名		説明
Application Server のインストールディレクトリー	-	Application Server のインストールディレクトリー
	common	Application Server - Base のインストールディレクトリー
	httpsd	Web Server のインストールディレクトリー
	javaee	Java EE Server のインストールディレクトリー
	jdk	Developer's Kit for Java のインストールディレクトリー

2.7 Application Server の運用管理について

Application Server の運用管理とは、ドメイン管理サーバを利用して Application Server を構成する各要素を管理し、Application Server のシステムを構築および運用することです。Application Server の運用管理には、コマンドと Administration Console を利用できます。

Application Server の運用管理では、Application Server を構成する Java EE サーバ、Web サーバおよびパフォーマンストレーサーを、ドメインという単位でグループ化して管理します。業務に応じて、構成の異なる Application Server をドメイン単位で管理できます。

Application Server の運用管理を実現するドメイン管理サーバとのインターフェースとして、コマンドと Administration Console が利用できます。コマンドと Administration Console のどちらを利用しても、Application Server の運用管理の各操作を実行できます。

コマンド

Application Server を構築および運用するための機能を、コマンドで実行するためのインターフェースです。コマンドの格納先をカレントディレクトリーにして、実行権限のあるユーザーが実行します。

Administration Console

Application Server を構築および運用するための機能を、Web ブラウザーから実行するためのインターフェースです。Web ブラウザーから Administration Console にログインして利用します。

2.8 Application Server V9 との互換性・移行性

Application Server V10 では、Application Server V9 で提供していた機能およびインターフェースについて、Java EE の標準仕様の範囲内では同様に使用できます。なお、V9 で提供していた製品独自機能について、V10 で使用できるのは一部の機能に限られます。

Application Server V9 が提供する機能の使用可否

Application Server V10 では、Java EE 7 に対応するために Java EE RI（GlassFish 4.1）を採用しています。Java EE の標準仕様の範囲内では V10 でも同様に使用できますが、Application Server V9 で提供していた製品独自機能について、V10 で使用できるのは一部の機能に限られます。

ユーザーインターフェースの互換性・移行性

ユーザーインターフェースについて、Application Server V9、Application Server V10 間の互換性を表に示します。なお、製品独自部分については、V9 から V10 への移行性を保証していません。

インターフェースの種別	互換性
標準仕様の API および DD	あり
製品独自の API、GUI、コマンド、ログフォーマット、および定義	なし

3 Application Server の設計項目

Application Server を利用したシステムを設計するために必要な情報について説明します。 システムの信頼性や性能を確保するために、必要に応じて設計項目を選択してシステムの設計をします。

3.1 Java のメモリー管理

Java のメモリー管理の設計で検討が必要なメモリー管理方式の仕組みや、設定するパラメーターについて説明しています。

3.1.1 Java のメモリー管理の方式について

Java のメモリー管理では、プログラムが使用するメモリー領域を、GC や Application Server の機能を使用して制御します。適切に Java のメモリー管理をするためには、使用するメモリー領域の構造や GC の処理の流れを理解する必要があります。Java のメモリー管理の方式として、SerialGC、SerialGC と明示管理ヒープ機能（GC を抑止するための機能）の組み合わせ、および G1GC という 3 種類から選択します。システムの要件に合わせた適切なメモリー管理の方式を選択することで、システム処理性能を向上できます。

Application Server および Application Server 上で動作するアプリケーションのメモリーは、GC を使用して管理します。GC の実行中にリクエストが送信された場合、GC が終了するまでリクエスト処理が停止するため、GC を適切に実行できるかどうかが、システムの処理性能に大きく影響します。

Application Server の Java のメモリー管理の方式を示します。メモリー管理の方式によって、GC の動作が異なるため、システム要件に合わせて Java のメモリー管理の方式を選択する必要があります。

項番	Java のメモリー管理の方式	適したシステムおよび特徴
1	SerialGC	スループットを重視するシステムに適しています。 • スループットが高い。 • GC の実行時間を制御できない。 • メモリーサイズのチューニングを実施することで、FullGC の発生を抑止できる。
2	SerialGC と明示管理ヒープ機能の組み合わせ	セッションを使用した一般的な Web フロントシステムに適しています。 • スループットが高い。 • GC の実行時間を制御できない。 • セッションを利用したシステムでは、メモリーサイズのチューニングのほか、明示管理ヒープ機能を使用してセッションを Explicit ヒープで管理することで、FullGC の発生を抑止できる。
3	G1GC	大規模なメモリーを使用するシステムやレスポンスを重視するシステムに適しています。 • スループットが低い。 • 一部の GC の実行時間を制御できる。 • メモリーサイズのチューニングをすることで、FullGC の発生を抑止できる。 • GC を実施するスレッド数を増やすことで、FullGC の発生を抑止できる。

関連項目

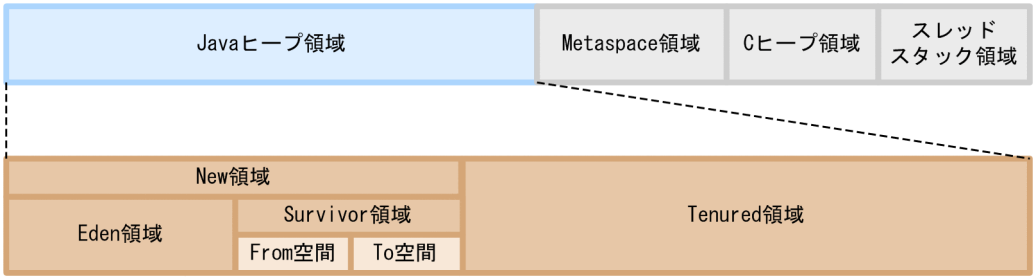
- 3.1.2　SerialGC のメモリー構造と GC の流れについて
- 3.1.3　SerialGC と明示管理ヒープ機能を組み合わせた場合のメモリー構造と GC の流れについて
- 3.1.4　G1GC のメモリー構造と GC の流れについて

3.1.2　SerialGC のメモリー構造と GC の流れについて

Java のメモリー管理の方式に SerialGC を選択すると、メモリー領域の使用状況に応じて、CopyGC や FullGC が発生します。SerialGC を選択する場合、-XX:+UseSerialGC オプションのほか、メモリー領域に関するオプションを指定します。

SerialGC のメモリー構造

Java のメモリー管理の方式に SerialGC を選択した場合の、メモリー構造を次の図に示します。

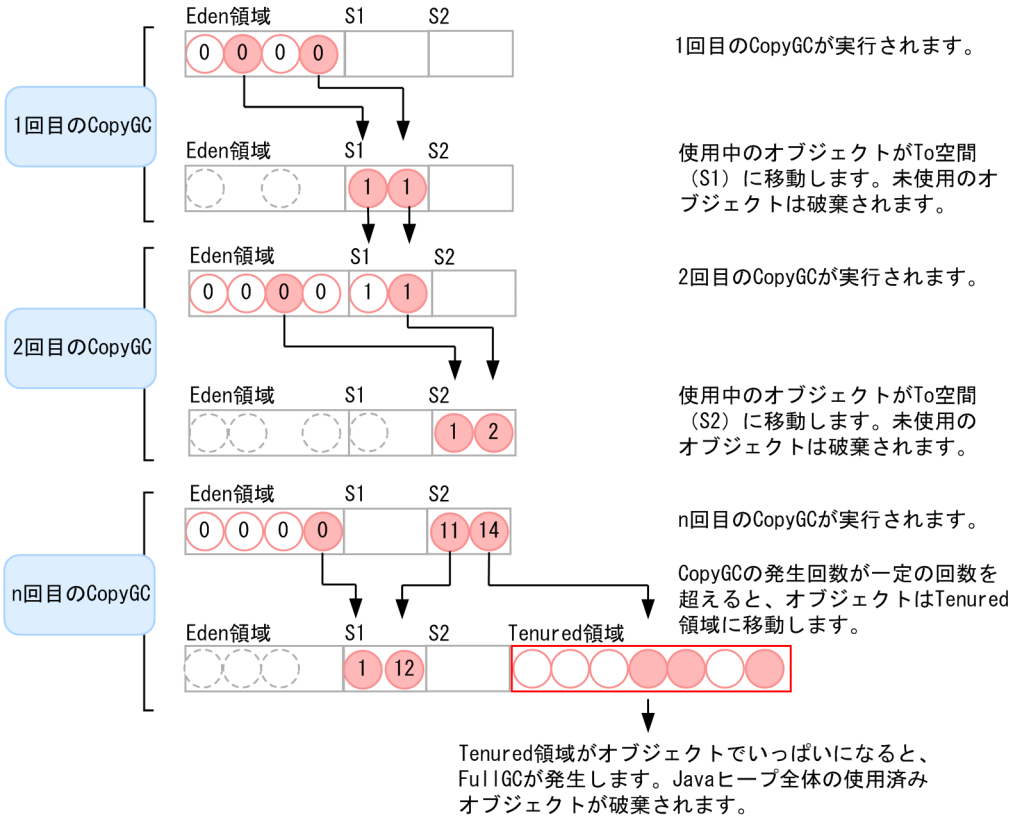


各領域の役割を次の表に示します。

項番	領域の名称	役割
1	Java ヒープ領域	Java プログラムが使用するメモリー領域です。大きく New 領域と Tenured 領域に分かれています。
2	New 領域	新しいオブジェクトが格納される領域です。Eden 領域と Survivor 領域に分かれています。
3	Eden 領域	作成直後のオブジェクトが格納される領域です。
4	Survivor 領域	New 領域に格納されていた Java オブジェクトのうち、CopyGC 実行時に破棄されなかった使用中のオブジェクトが格納される領域です。From 空間と To 空間に分かれています。
5	Tenured 領域	長時間使用するオブジェクトが格納される領域です。
6	Metaspace 領域	ロードされたクラス情報やメソッド情報が格納される領域です。
7	C ヒープ領域	Developer's Kit for Java がネイティブライブラリーを実行する際に使用する領域です。OS 固有領域です。
8	スレッドスタック領域	スレッドごとに保持するスタック領域です。OS 固有領域です。

SerialGC の場合の GC の流れ

SerialGC の場合の GC の流れを示します。



（凡例）

x ：使用中のオブジェクト（xはCopyGCの対象となった回数）

x ：使用済み（未使用）のオブジェクト（xはCopyGCの対象となった回数）

：破棄されたオブジェクト

S1、S2：Survivor領域のFrom空間とTo空間を指します。CopyGCのたびに位置が入れ換わります。

1. 1 回目の CopyGC

   生成されたオブジェクトは、 New 領域の Eden 領域に格納されます。Eden 領域にオブジェクトがいっぱいになると、1 回目の CopyGC が発生します。CopyGC は New 領域全体を対象としています。CopyGC が発生すると、使用済みのオブジェクトを削除し、使用中のオブジェクトを Survivor 領域の To 空間（S1）に移動します。

2. 2 回目の CopyGC

   1 回目の CopyGC 発生後、再び Eden 領域にオブジェクトがいっぱいになると、2 回目の CopyGC が発生します。CopyGC は New 領域全体を対象とするため、1 回目の CopyGC で Eden 領域から Survivor 領域の To 空間（S1）に移動したオブジェクトも CopyGC の対象となります。2 回目の CopyGC 発生時に Eden 領域に格納されていた使用中のオブジェクトは、To 空間（S2）に移動し、同じく From 空間（S1）にある使用中のオブジェクトも To 空間（S2）に移動します。

3. n 回目の CopyGC

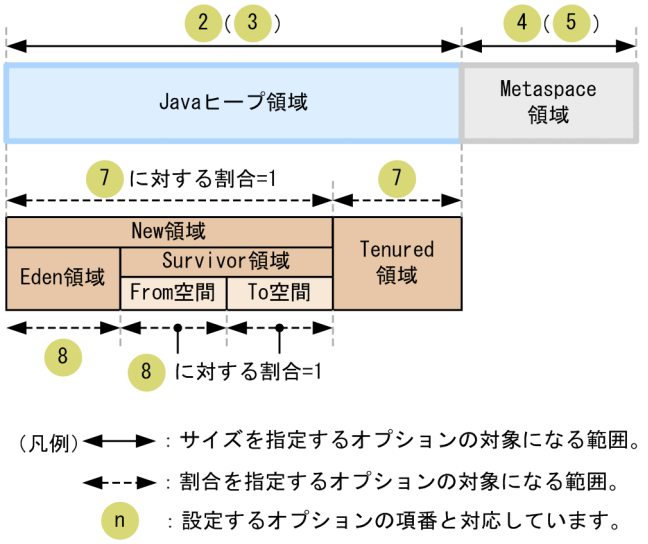
Survivor 領域に移動したオブジェクトは、CopyGC 発生時に使用中の場合、Survivor 領域内で From 空間と To 空間を交互に移動し続けます。あるオブジェクトについて CopyGC の発生回数が一定の回数を超えると（図の場合 15 回）、そのオブジェクトは Survivor 領域から Tenured 領域に移動します。

4. FullGC

   Tenured 領域がオブジェクトでいっぱいになると FullGC が発生し、Java ヒープ領域全体を対象に使用済みオブジェクトを削除します。

SerialGC の場合に設定するパラメーター

SerialGC の場合、長時間アプリケーションを停止する FullGC が必ず発生するため、FullGC を発生できるタイミングまで FullGC が発生しないようにメモリーの設計・チューニングをする必要があります。SerialGC の場合に設定するパラメーターを示します。



項番	項目	オプション名	説明
1	SerialGC の設定	-XX:+UseSerialGC	メモリー管理方式に SerialGC を選択します。このパラメーターは、デフォルトで有効です。
2	メモリーの設計	-Xmx*Javaヒープ領域の最大サイズ*	Java ヒープ領域の最大サイズを設定します。
3		-Xms*Javaヒープ領域の初期サイズ*	Java ヒープ領域の初期サイズを設定します。このパラメーターは、-Xmx オプションと同じ値の設定を推奨します。
4		-XX:MaxMetaspaceSize=*Metaspace領域の最大サイズ*	Metaspace 領域の最大サイズを設定します。
5		-XX:MetaspaceSize=*Metaspace領域に起因するFullGC の基準値*	Metaspace 領域に起因する FullGC の基準値を設定します。Metaspace 領域のサイズが基準値を超えると FullGC が発生するため、アプリケーションで必要なクラス情報のサイズから見積もった値を設定します。このパラメーターは、-XX:MaxMetaspaceSize オプションと同じ値の設定を推奨します。

項番	項目	オプション名	説明
6		-XX:CompressedClassSpaceSize=*Compressed Class Spaceの最大サイズ*	圧縮オブジェクトポインター機能が有効な場合に Metaspace 領域内に作成される Compressed Class Space 領域の最大サイズを設定します。このパラメーターは、-XX:MaxMetaspaceSize オプションと同じ値の設定を推奨します。
7		-XX:NewRatio=*New領域に対するTenured領域の割合*	New 領域を 1 とした場合の Tenured 領域の割合を設定します。
8		-XX:SurvivorRatio=*Survivor領域のFrom空間とTo空間に対するEden領域の割合*	Survivor 領域の From 空間と To 空間を 1 とした場合の、Eden 領域の割合を設定します。

関連項目

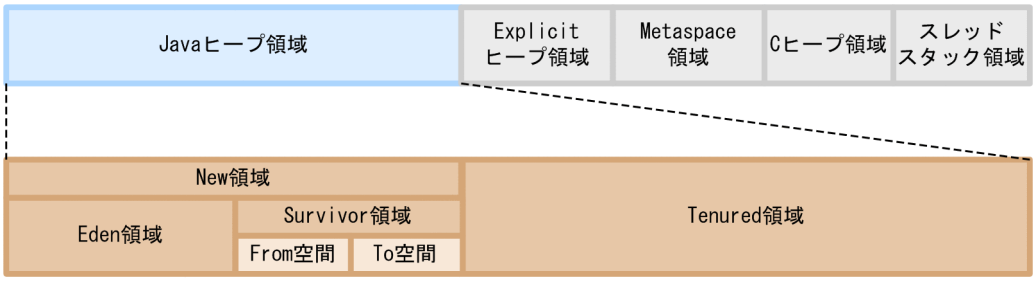
- 4.6.5　create-jvm-options サブコマンドを使用して JavaVM オプションを変更する
- 8.2.3　create-jvm-options サブコマンドを使用して JavaVM オプションを変更する

3.1.3　SerialGC と明示管理ヒープ機能を組み合わせた場合のメモリー構造と GC の流れについて

Java のメモリー管理の方式に、SerialGC と明示管理ヒープ機能を組み合わせた方式を選択すると、SerialGC で発生する FullGC を明示管理ヒープ機能で抑止できます。SerialGC と明示管理ヒープ機能を組み合わせた方式を選択する場合、-XX:+UseSerialGC オプションのほか、メモリーに関するオプション、明示管理ヒープ機能の設定や Explicit ヒープ領域のメモリー設計に関するオプションを指定します。

SerialGC と明示管理ヒープ機能を組み合わせた場合のメモリー構造

Java のメモリー管理の方式に SerialGC と明示管理ヒープ機能を組み合わせた場合の、メモリー構造を次の図に示します。



各領域の役割を次の表に示します。

項番	領域の名称	役割
1	Java ヒープ領域	Java プログラムが使用するメモリー領域です。大きく New 領域と Tenured 領域に分かれています。
2	New 領域	新しいオブジェクトを格納する領域です。Eden 領域と Survivor 領域に分かれています。
3	Eden 領域	作成直後のオブジェクトを格納する領域です。
4	Survivor 領域	1 回以上 GC を実施した、使用中のオブジェクトを格納する領域です。
5	Tenured 領域	長時間使用するオブジェクトを格納する領域です。
7	Explicit ヒープ領域	セッションオブジェクトなどを格納する領域です。
6	Metaspace 領域	ロードされたクラス情報やメソッド情報を格納する領域です。
8	C ヒープ領域	Developer's Kit for Java がネイティブライブラリーを実行する際に使用する領域です。
9	スレッドスタック領域	スレッドごとに保持するスタック領域です。

SerialGC と明示管理ヒープ機能を組み合わせた場合のメモリー管理の仕組み

メモリー管理方式に SerialGC を選択し、Java ヒープ領域を使用していくと FullGC が発生することがあります。FullGC の発生を抑えるためには、明示管理ヒープ機能を使用して、Java ヒープ領域とは別の Explicit ヒープ領域に、FullGC の要因となるオブジェクトを格納します。FullGC の対象となる Tenured 領域に、FullGC の要因となるオブジェクトが格納されないため、FullGC の発生を抑止できます。なお、Explicit ヒープ領域に格納されたオブジェクトは、寿命が切れたタイミングで明示的に解放されます。

FullGC の要因となるオブジェクト

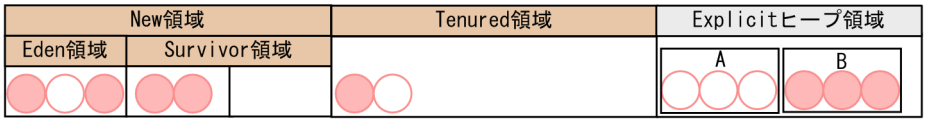
長寿命オブジェクトで、かつ、一定期間が経過したあと不要になるオブジェクトが、FullGC の要因となります。このようなオブジェクトには、例えば、ログインからログアウトまでの一連の処理で使用するセッション情報に関係するオブジェクトが該当します。セッション情報は複数のリクエストにわたって使用するため、長寿命で、一定期間が経過してログアウトすると不要になります。Application Server では、セッション情報に関係するオブジェクトは、Explicit ヒープ領域に格納するオブジェクトとしてデフォルトで設定されています。

オブジェクトが不要になったときのメモリーの状態

寿命がわかっているオブジェクトが、一定期間が経過したあと不要になったときのメモリーの状態を次の図に示します。なお、明示管理ヒープ機能を使用した場合の Explicit ヒープ領域は、Explicit メモリーブロックという、複数のメモリーブロックで構成されています。この Explicit メモリーブロック単位で、Explicit 領域内のメモリーの初期化や解放が行われます。

明示管理ヒープ機能を使用した場合

Explicit ヒープ内の寿命がわかっているオブジェクト群 A が、一定期間が経過したあと不要になると、オブジェクト群 A が格納されている Explicit メモリーブロックがブロックごと削除されます。

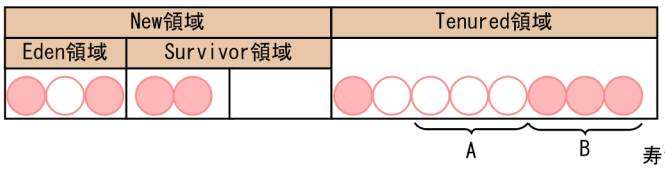


寿命がわかっているオブジェクト群Aが不要になると、Explicitメモリーブロックごと削除されます。

（凡例）
- ：使用中のオブジェクト
- ：使用済み（未使用）のオブジェクト
- ：Explicitメモリーブロック
- A ：寿命がわかっているオブジェクト群A
- B ：寿命がわかっているオブジェクト群B

明示管理ヒープ機能を使用しない場合

Tenured 領域内の寿命がわかっているオブジェクト群 A が、一定期間が経過したあと不要になると、FullGC が発生するまで Tenured 領域に残ります。

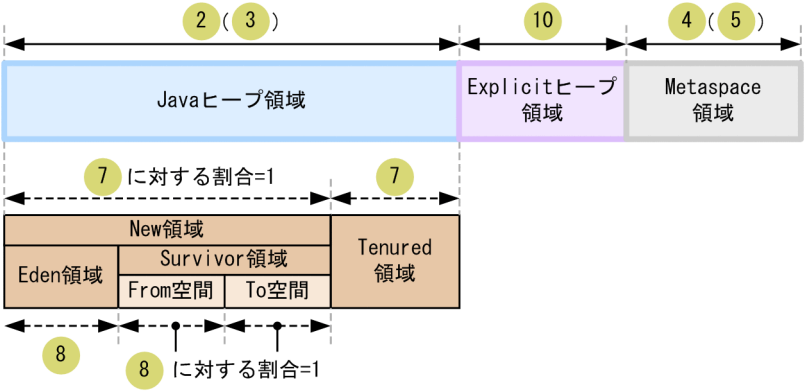


寿命がわかっているオブジェクト群Aが不要になると、FullGCが発生するまでTenured領域に残ります。

（凡例）
- ：使用中のオブジェクト
- ：使用済み（未使用）のオブジェクト
- A ：寿命がわかっているオブジェクト群A
- B ：寿命がわかっているオブジェクト群B

SerialGC と明示管理ヒープ機能を組み合わせた場合に設定するパラメーター

メモリー管理方式に SerialGC と明示管理ヒープ機能を組み合わせた方式を選択する場合、Java ヒープ領域と Explicit ヒープ領域のメモリーの設計およびチューニングをする必要があります。SerialGC と明示管理ヒープ機能を組み合わせた場合に設定するパラメーターを示します。



（凡例）
- ：サイズを指定するオプションの対象になる範囲。
- ：割合を指定するオプションの対象になる範囲。
- n ：設定するオプションの項番と対応しています。

項番	項目	オプション名	説明
1	SerialGC の設定	-XX:+UseSerialGC	メモリー管理方式に SerialGC を選択します。このパラメーターは、デフォルトで有効です。
2	メモリーの設計	-Xmx*Javaヒープ領域の最大サイズ*	Java ヒープ領域の最大サイズを設定します。
3		-Xms*Javaヒープ領域の初期サイズ*	Java ヒープ領域の初期サイズを設定します。このパラメーターは、-Xmx オプションと同じ値の設定を推奨します。
4		-XX:MaxMetaspaceSize=*Metaspace領域の最大サイズ*	Metaspace 領域の最大サイズを設定します。
5		-XX:MetaspaceSize=*Metaspace領域に起因するFullGC の基準値*	Metaspace 領域に起因する FullGC の基準値を設定します。Metaspace 領域のサイズが基準値を超えると FullGC が発生するため、アプリケーションで必要なクラス情報のサイズから見積もった値を設定します。このパラメーターは、-XX:MaxMetaspaceSize オプションと同じ値の設定を推奨します。
6		-XX:CompressedClassSpaceSize=*Compressed Class Spaceの最大サイズ*	圧縮オブジェクトポインター機能が有効な場合に Metaspace 領域内に作成される Compressed Class Space 領域の最大サイズを設定します。このパラメーターは、-XX:MaxMetaspaceSize オプションと同じ値の設定を推奨します。
7		-XX:NewRatio=*New領域に対するTenured領域の割合*	New 領域を 1 とした場合の Tenured 領域の割合を設定します。
8		-XX:SurvivorRatio=*Survivor領域のFrom空間とTo空間に対するEden領域の割合*	Survivor 領域の From 空間と To 空間を 1 とした場合の、Eden 領域の割合を設定します。
9	明示管理ヒープ機能の設定	-XX:+HitachiUseExplicitMemory	明示管理ヒープ機能を有効にします。このパラメーターは、デフォルトで有効です。
10	Explicit ヒープ領域のメモリー設計	-XX:HitachiExplicitHeapMaxSize=*Explicitヒープ領域の最大サイズ*	Explicit ヒープ領域の最大サイズを設定します。

関連項目

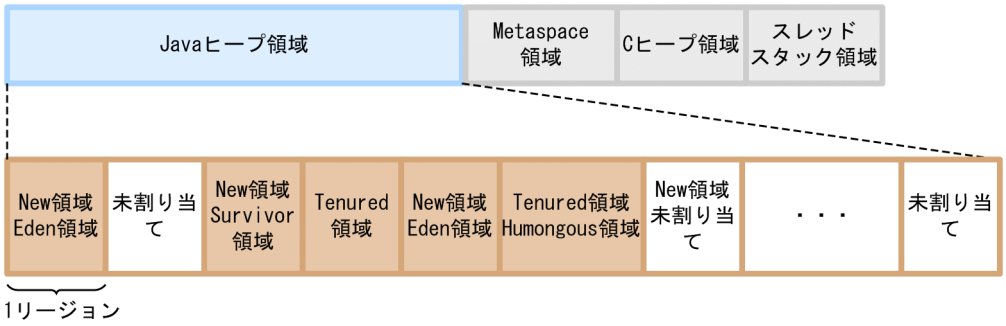
- 4.6.5　create-jvm-options サブコマンドを使用して JavaVM オプションを変更する
- 8.2.3　create-jvm-options サブコマンドを使用して JavaVM オプションを変更する

3.1.4　G1GC のメモリー構造と GC の流れについて

Java のメモリー管理の方式に G1GC を選択すると、メモリー領域に対して YoungGC、MixedGC、および FullGC の 3 種類の GC が発生します。このうち、YoungGC と MixedGC については、GC 発生時にアプリケーションが停止する時間を制御できます。G1GC を選択する場合、-XX:+UseG1GC オプションのほか、GC 発生時の目標の停止時間、メモリーに関するオプションや処理性能のチューニングに関するオプションを指定します。

G1GC のメモリー構造

Java のメモリー管理の方式に G1GC を選択した場合の、メモリー構造を次の図に示します。



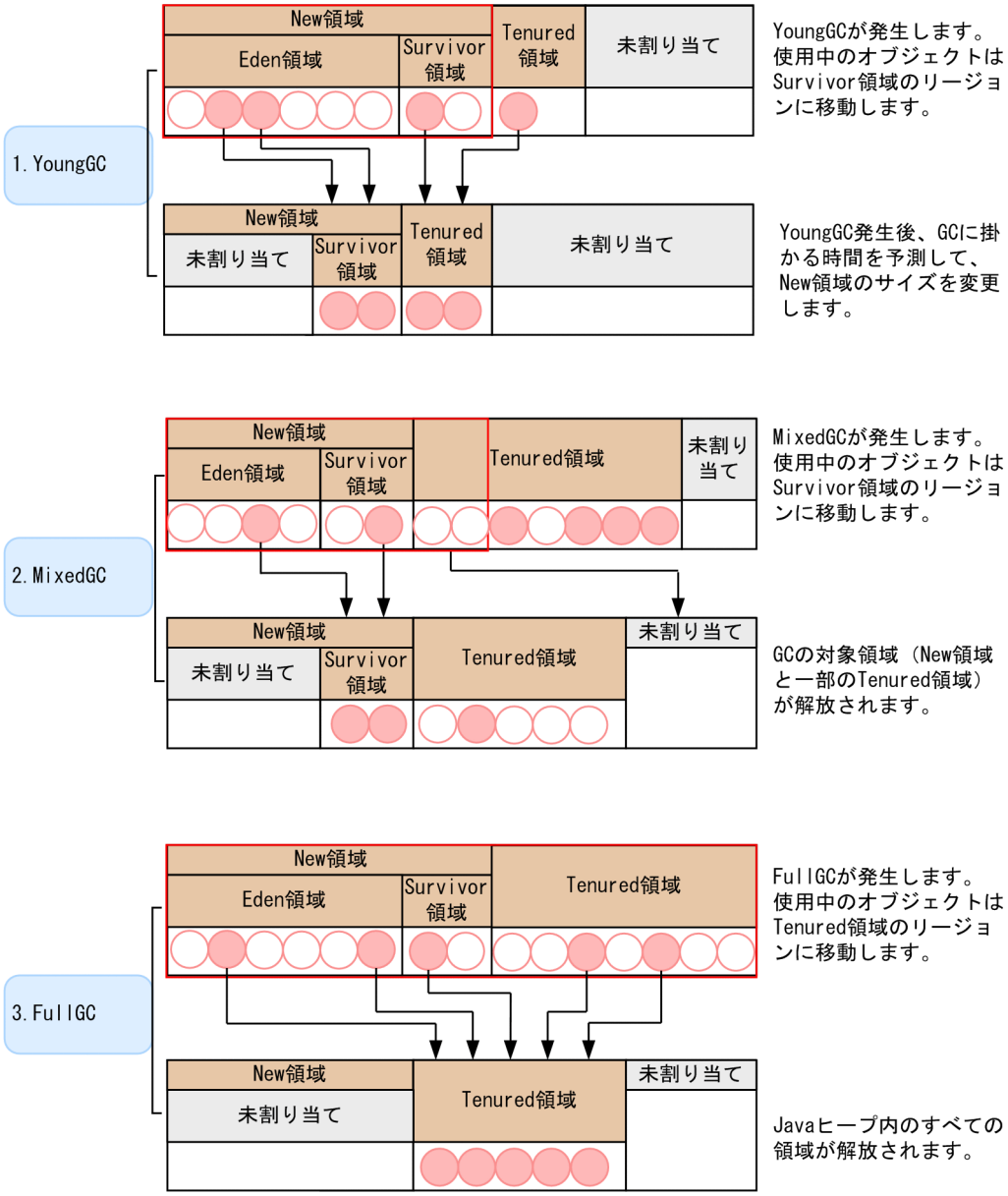
各領域の役割を次の表に示します。

項番	領域の名称	役割
1	Java ヒープ領域	Java プログラムが使用するメモリー領域です。大きく New 領域と Tenured 領域に分かれています。
2	New 領域	新しいオブジェクトを格納する領域です。Eden 領域と Survivor 領域に分かれています。
3	Eden 領域	作成直後のオブジェクトを格納する領域です。
4	Survivor 領域	1 回以上 GC を実施した、使用中のオブジェクトを格納する領域です。
5	Tenured 領域	長時間使用するオブジェクトを格納する領域です。
6	Metaspace 領域	ロードされたクラス情報やメソッド情報を格納する領域です。
7	C ヒープ領域	Developer's Kit for Java がネイティブライブラリーを実行する際に使用する領域です。
8	スレッドスタック領域	スレッドごとに保持するスタック領域です。
9	New 領域の未割り当ての領域	New 領域内で、Eden 領域または Survivor 領域に割り当てていない領域です。
10	Humongous 領域	サイズの大きいオブジェクトを格納する領域です。Tenured 領域の一部で、連続したリージョンを割り当てます。
11	未割り当ての領域	Java ヒープ領域内で、どの領域にも割り当てていない領域です。

G1GC の場合の GC の流れ

G1GC では、Java ヒープをリージョンというメモリーブロック単位で管理しているため、New 領域や Tenured 領域を連続領域として確保していません。オブジェクトはリージョン内に格納し、リージョンに空きがない場合は、未割り当てのリージョンを領域に割り当て、オブジェクトを格納します。

メモリー管理方式に G1GC を選択した場合の GC の流れを次の図で説明します。



1. YoungGC

New 領域に割り当てたリージョンに空きがなくなると、YoungGC が発生します。YoungGC が発生すると、使用中のオブジェクトは Survivor 領域に割り当てたリージョンに移動し、使用済みのオブジェクトはリージョンごと解放します。また、YoungGC 発生時に使用中のオブジェクトは、Survivor 領域に割り当てたリージョン間を移動し続け、ある一定の回数移動すると Tenured 領域に割り当てたリージョンに移動します。YoungGC 発生後は、GC に掛かった時間から、次の GC に掛かる時間を予測して、New 領域のサイズを変更します。YoungGC 発生後の図は、予測した時間より長く GC に時間が掛かったために、New 領域を縮小した場合の例です。

2. MixedGC

Tenured 領域の使用率が増加すると、MixedGC が発生します。MixedGC が発生すると、New 領域に割り当てたリージョンに加えて、目標停止時間内に収まる範囲で、一部の Tenured 領域に割り当てたリージョンが GC の対象となります。この一部の Tenured 領域に割り当てたリージョンは、アプリケーションの実行と並行して行っているオブジェクトが使用中かどうかの解析情報に基づいて、解放されるサイズが大きいと予測されるリージョンから優先して GC の対象となります。そのため、オブジェクトの情報解析が十分に実施されていない場合や、解析の結果 MixedGC の効果が低い場合は、MixedGC は発生しません。

3. FullGC

Java ヒープ内のリージョンに空きがなくなり、MixedGC が発生しない場合、Java ヒープ全体を対象として FullGC が発生します。

G1GC の場合に設定するパラメーター

メモリー管理方式に G1GC を選択した場合、GC によるアプリケーションの停止時間とスループットがシステム要件を満たすように、メモリーの設計やチューニングをする必要があります。なお、G1GC は FullGC の発生によるアプリケーションの停止時間を制御できません。そのため、FullGC が発生している場合、FullGC が発生しないようにチューニングをする必要があります。G1GC の場合に設定するパラメーターを示します。

項番	項目	オプション名	説明
1	G1GC の設定	-XX:+UseG1GC	メモリー管理方式に G1GC を選択します。
2		-XX:MaxGCPauseMillis=*目標停止時間*	目標停止時間をミリ秒単位で設定します。
3	メモリーの設計	-Xmx*Javaヒープ領域の最大サイズ*	Java ヒープ領域の最大サイズを設定します。
4		-Xms*Javaヒープ領域の初期サイズ*	Java ヒープ領域の初期サイズを設定します。このパラメーターは、-Xmx オプションと同じ値の設定を推奨します。
5		-XX:MaxMetaspaceSize=*Metaspace領域の最大サイズ*	Metaspace 領域の最大サイズを設定します。
6		-XX:MetaspaceSize=*Metaspace領域に起因するFullGC の基準値*	Metaspace 領域に起因する FullGC の基準値を設定します。Metaspace 領域のサイズが基準値を超えると FullGC が発生するため、アプリケーションで必要なクラス情報のサイズから見積もった値を設定します。このパラメーターは、-XX:MaxMetaspaceSize オプションと同じ値の設定を推奨します。
7		-XX:CompressedClassSpaceSize=*Compressed Class Spaceの最大サイズ*	圧縮オブジェクトポインター機能が有効な場合に Metaspace 領域内に作成される Compressed Class Space 領域の最大サイズを設定します。このパラメーターは、-XX:MaxMetaspaceSize オプションと同じ値の設定を推奨します。
8		-XX:SurvivorRatio=*New領域に対するSurvivor領域が最大の場合の割合*	New 領域に対する Survivor 領域が最大サイズの場合の割合を設定します。
9	処理性能のチューニング	-XX:ParallelGCThreads=*YoungGCおよびMixedGCを行うスレッド数*	YoungGC および MixedGC を行うスレッド数を設定します。-XX:ConcGCThreads のスレッド数に合わせて設定します。

項番	項目	オプション名	説明
10		-XX:ConcGCThreads=*アプリケーションと並行して処理を実行するスレッド数*	アプリケーションと並行して処理を実行するスレッド数を設定します。

関連項目

- 4.6.5　create-jvm-options サブコマンドを使用して JavaVM オプションを変更する
- 8.2.3　create-jvm-options サブコマンドを使用して JavaVM オプションを変更する

3.2　負荷分散について

負荷分散とは、Application Server で処理するリクエストを複数の Application Server に分散させて、リクエスト処理に掛かる負荷を分散させることです。負荷分散をするには、ハードウェアロードバランサーまたはソフトウェアロードバランサーを使用します。

負荷分散の手段の選択

負荷分散には、ハードウェアロードバランサーまたはソフトウェアロードバランサーのどちらかを使用します。Application Server では、ソフトウェアロードバランサーとして Web Server を使用した負荷分散機能を提供しています。この負荷分散機能では、リクエスト数を基に各サーバに処理を振り分けます。

ハードウェアロードバランサーまたはソフトウェアロードバランサーを導入する際の、コストおよび機能性に関する比較を次の表に示します。

負荷分散の手段	コスト	機能性
ハードウェアロードバランサー	高	高
ソフトウェアロードバランサー	低	低

関連項目

- 5.1　システム周辺環境の設定について
- 5.2　ソフトウェアロードバランサーを設定する

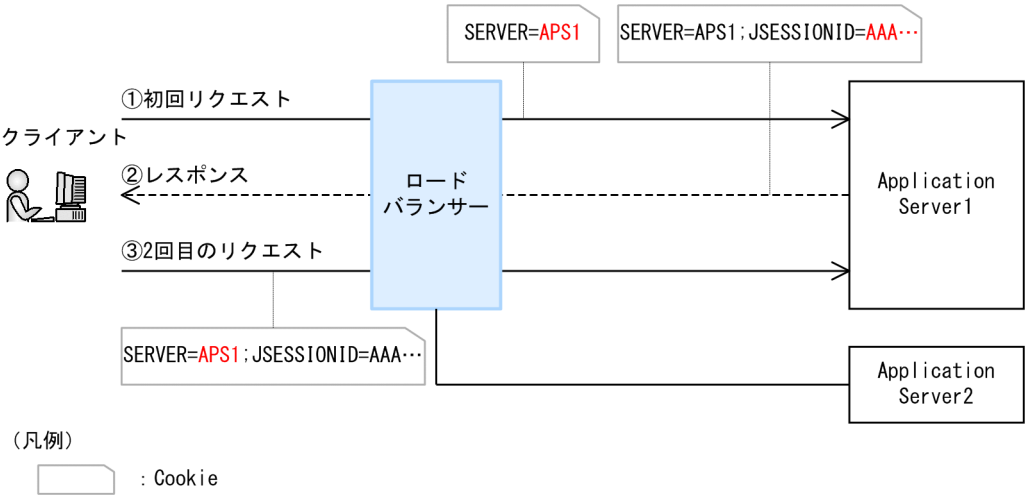
3.3　セッション管理について

セッション管理とは、リクエストと Web クライアントの対応付けをサーバ ID とセッション ID を使って管理することです。業務システムの利用者がログインからログアウトまでの一連の業務処理を行う場合に、リクエストと業務情報を結びつける仕掛けをセッションと呼びます。セッション管理をすることで、ロードバランサーを使用したシステムでもセッションを継続できます。

セッション管理の仕組み

Application Server では、セッション中の一連のリクエストに対して、同一のセッション ID を Cookie やクエリーストリングに付与して、セッションを継続させています。

ロードバランサーを配置するシステム構成で、セッションを維持する仕組みを次の図で説明します。



1. クライアントからの初回リクエスト

   ロードバランサーが、クライアントから初回リクエストを受け付けると、ロードバランサーは Application Server1 に対してリクエストを転送します。このとき、ロードバランサーは、サーバ識別用の Cookie をリクエストに含めて Application Server1 に送付します。

2. 初回リクエストに対するレスポンス

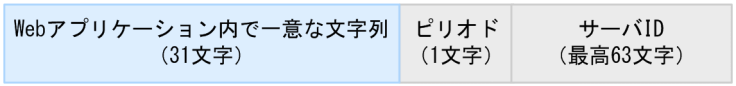
   Application Server1 は、サーバ識別用の Cookie とセッション ID をレスポンスに含めて、クライアントに返信します。

3. クライアントからの 2 回目のリクエスト

   2 回目のリクエストには、サーバ識別用の Cookie を含めてロードバランサーに送付します。2 回目のリクエストを受け付けたロードバランサーは、リクエストに含まれたサーバ識別用の Cookie からリクエスト転送先を判別し、セッションを確立しているサーバ（Application Server1）にリクエストを送付します。

セッション ID の構成

Application Server では、セッションの管理に使用するセッション ID を一意にするため、セッション ID に対してサーバ ID を付与しています。セッション ID の構成を次の図に示します。



セッション ID の 33 文字目以降に、Java EE サーバごとに異なる識別子であるサーバ ID を含めることで、システム内に存在する複数の Java EE サーバ間で セッション ID が一意であることを保証します。

3.4　トランザクション管理について

データベースやトランザクションサーバなどの企業内に構築されているバックエンドシステム（EIS）で、業務取引でのデータの一貫性を保証するためにはトランザクション管理が必要です。Application Server のトランザクションでは、ローカルトランザクションを利用することを推奨します。

トランザクション管理の仕組み

Application Server で利用できるトランザクション管理の方式には、ローカルトランザクションとグローバルトランザクションがあります。システム統合などでグローバルトランザクションを使用する必要がある場合を除いて、ローカルトランザクションを使用することを推奨します。

ローカルトランザクション

単一のリソースが参加するトランザクションです。

グローバルトランザクション

複数のリソースが参加するトランザクションです。グローバルトランザクションは、複数のリソースを同期して制御する必要があります。なお、同期処理の制御にはコストが掛かります。

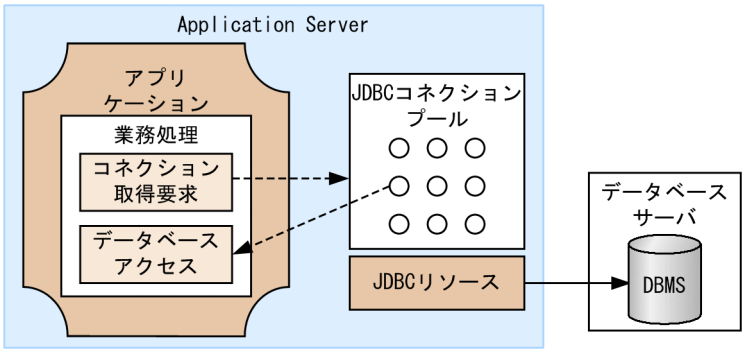
3.5　コネクション管理について

コネクションの管理とは、JDBC コネクションプールを利用して、データベースサーバに対するコネクションを管理することです。JDBC コネクションプールにプールするコネクション数を管理することで、データベースアクセスの処理時間を低減できます。また、コネクション障害検知機能を使用することで、JDBC コネクションプールのコネクションに障害が発生した場合でも、処理を続けることができます。

コネクション管理の仕組み

データベースサーバへの接続で使用するコネクションの生成は時間が掛かる処理です。そこで、JDBC コネクションプールに一定数のコネクションをプールしておきます。コネクションの取得要求を受け取ると、JDBC コネクションプールにプールしていたコネクションを返して、コネクションを使い回すことで、データベースアクセスに掛かる処理時間を低減します。

コネクション管理の仕組みを次の図に示します。

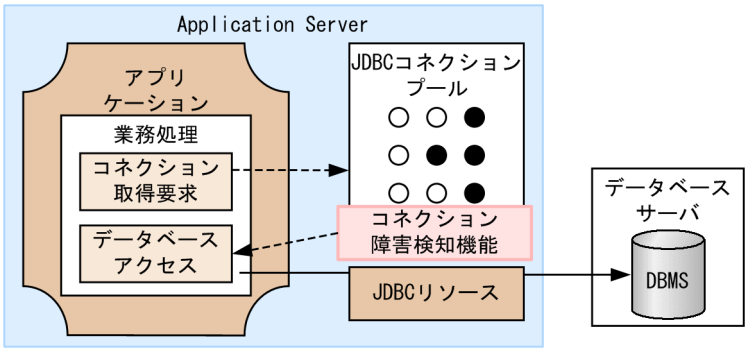


（凡例）
○：コネクション

コネクション障害検知機能

コネクション障害検知機能では、コネクションの取得要求を受け取ると、JDBC コネクションプールのコネクションに障害が発生していないかをチェックして、障害が発生していないコネクションを返します。これによって、リソースダウンやネットワーク障害などでコネクションに障害が発生した場合でも、障害が発生していないコネクションを利用して処理を続けることができます。

コネクション障害検知機能を次の図に示します。



（凡例）

○：障害が発生していないコネクション

●：障害が発生しているコネクション

コネクションプールのコネクション数の最大値が多い場合、メモリを多く必要とします。

サーバ障害やネットワーク障害などが多発した場合、コネクション障害検知を行う際に必要なリソースが不足し、コネクション取得要求がエラーとなります。この場合、KDKD20004-W のメッセージが出力されます。また、コネクション障害検知でタイムアウトが発生してコネクションを破棄する場合も、障害が多発してコネクションの破棄に必要なリソースが不足すると、コネクションの破棄で応答が返らなくなるおそれがあります。この場合、KDKD20001-W のメッセージが出力されます。

なお、コネクション障害検知機能を使用する場合、障害が発生したコネクションを破棄する前に新たにコネクションを確立することで、DBMS の最大同時接続数が一時的にコネクションプールのサイズを超えることがあります。

JDBC コネクションプールのチューニング

コネクションプールを作成する際、物理コネクションの接続先データベースで設定されているコネクション数がコネクションプールの最大サイズ以上となるように、値をチューニングする必要があります。

接続先データベースで設定されているコネクション数については、各データベースのマニュアルを参照してください。

DBMS の最大同時接続数の見積もり式

DBMS の最大同時接続数の見積もり式を示します。JDBC コネクションプールを複数利用する場合の最大同時接続数は、この式の総和になります。

DBMSの最大同時接続数＝*JDBCコネクションプールのサイズ*＋α※

注※

＋α は、DBMS の最大同時接続数が、一時的に JDBC コネクションプールのサイズを超えるおそれがあることを示します。コネクション障害検知機能を使用する場合、障害が発生したコネクションを破棄する前に新たにコネクションを確立することで、DBMS の最大同時接続数が一時的にコネクションプー

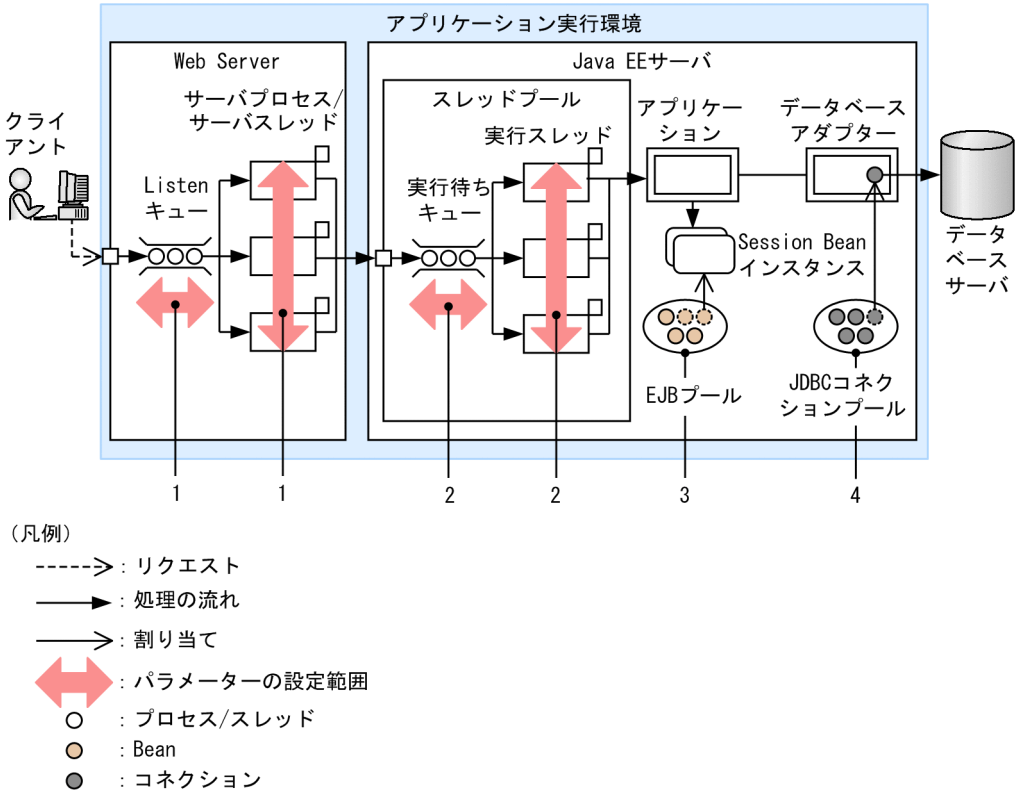
ルのサイズを超えることがあります。なお、＋α は、最大でコネクションプールのサイズと同じになります。

3.6 流量制御について

流量制御とは、リクエスト処理が集中した状態でもシステムを安定して稼働させるために、システムで処理するリクエストの数を制御することです。Java EE サーバで一度に実行されるリクエスト処理数の最大値を制御したり、EJB プールにプールする Bean 数や JDBC コネクションプールにプールする JDBC コネクション数を制御したりすることで、Java EE サーバの負荷を一定に抑え、安定したスループットを実現できます。

流量制御の仕組み

リクエスト処理の流れと流量制御の対象について次の図に示します。



制御対象リソースと設定するパラメーターについて次の表に示します。図中の項番と表の項番が対応しています。

項番	制御対象リソース	制御内容	設定するパラメーター
1	Web Server サーバプロセス/サーバスレッド	Web Server のサーバプロセス/サーバスレッドは、主に Java EE サーバにリクエストを転送する役割を持つプロセスです。このプロセスは Java EE サーバでのリクエスト処理が完了するまで占有されます。 最大プロセス数を変更して、最大同時接続数を制御します。1 つのリクエストに対して、プロセスが 1 つ割り当てられます。また、処理しきれないリクエストを格納するキューのサイズも変更できます。	• キューのサイズ：ListenBackLog • 最大同時接続数：MaxRequestWorkers • サーバプロセスの稼働数：HWSKeepStartServers、MinSpareServers、MaxSpareServers、StartServers

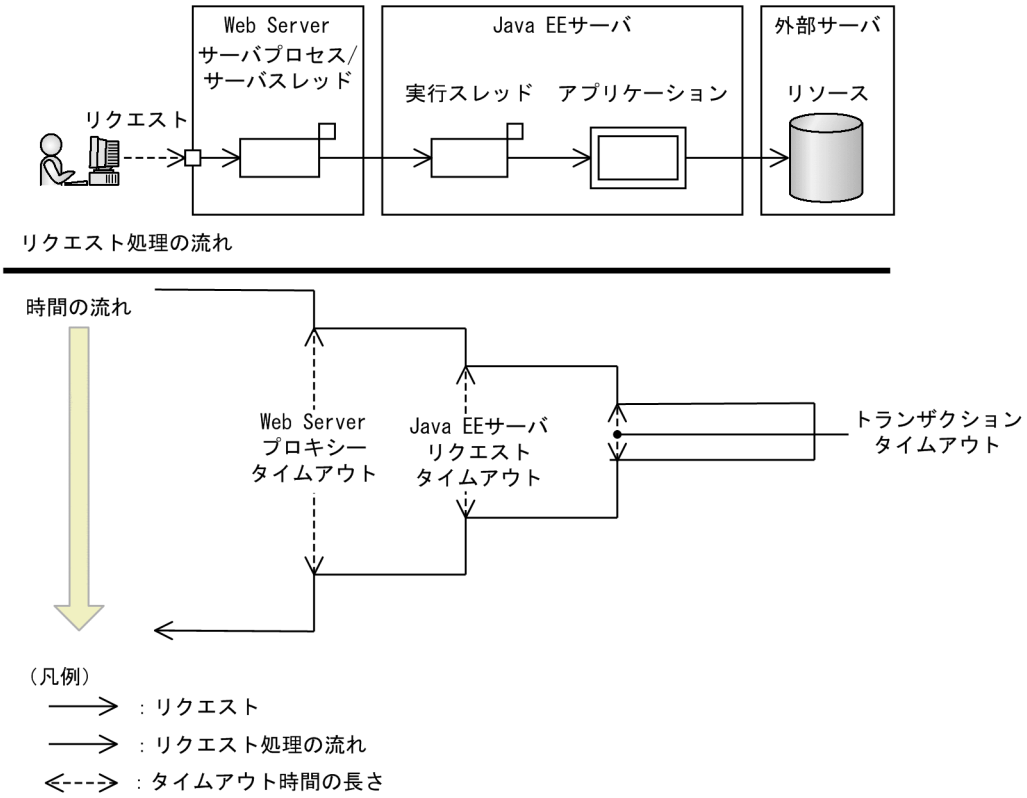
項番	制御対象リソース	制御内容	設定するパラメーター
2	Java EE サーバ実行スレッドプール	Java EE サーバ実行スレッドは、リクエストの解釈、Web アプリケーションの実行、およびレスポンスの構築をするスレッドです。 最大スレッド数を変更して、最大同時実行リクエスト数を制御します。1 つのリクエストに対して、スレッドが 1 つ割り当てられます。また、処理しきれないリクエストを格納するキューのサイズも変更できます。	• 最大実行スレッド数：configs.config.*Java EEサーバの構成名*.thread-pools.thread-pool.*スレッドプールID*.max-thread-pool-size • 実行待ちキューサイズ：configs.config.*Java EEサーバの構成名*.thread-pools.thread-pool.*スレッドプールID*.max-queue-size
3	EJB プール	プールする最大の Bean 数を変更して、同時に実行するスレッド数を制御します。1 つのリクエストに対して、Session Bean が 1 つ割り当てられます。 なお、利用している Bean 数が最大の場合に新たな Bean が必要になったとき、クライアントは Bean が利用できるようになるまで待つ必要があるため、リクエストが滞留します。	最大プールサイズ：configs.config.*Java EEサーバの構成名*.ejb-container.max-pool-size
4	JDBC コネクションプール	プールする最大の JDBC コネクション数を制御します。 なお、利用しているコネクション数が最大の場合に新たなコネクションが必要になったとき、クライアントはコネクションが利用できるようになるまで待つ必要があるため、リクエストが滞留します。	最大プールサイズ：resources.jdbc-connection-pool.*コネクションプールID*.max-pool-size

3.7　タイムアウト制御について

タイムアウト制御とは、処理に掛かる時間のタイムアウトを設定することです。タイムアウトを設定すると、設定時間を超えた処理は中断されるため、システム構築者にとって意図しないリソースの占有を防いだり、業務システムのユーザーに対してシステムの無反応を回避したりできます。

タイムアウト制御の仕組み

リクエスト処理の流れとタイムアウト時間の関係について、次の図に示します。



この図の呼び出し関係はすべて同期のため、実行時間はクライアントに近いほど長くなります。このため、タイムアウト個所を特定しやすいように、呼び出し元のタイムアウトが、呼び出し先のタイムアウトよりも長くなるように設定してください（Web Server プロキシータイムアウト＞ Java EE サーバリクエストタイムアウト＞トランザクションタイムアウト）。

このようなケースのほか、JAX-RS や JAX-WS を使用して、複数のアプリケーションを連携させる場合も、呼び出し元および呼び出し先で、接続タイムアウトおよびリクエストタイムアウトのタイムアウト時間を設定できます。

Application Server で設定できる、タイムアウトについて説明します。

サーバ処理のタイムアウト

サーバ処理のタイムアウトの制御内容、設定するパラメーターを次の表に示します。

項番	設定項目名	制御内容	制御対象	設定するパラメーター
1	Web Server タイムアウト	クライアントとのデータ送受信の待ち時間	Web Server サーバプロセス/サーバスレッド	Timeout
2	Web Server プロキシータイムアウト	Java EE サーバへリクエストを送信して、レスポンスを受け取るまでの時間	Web Server サーバプロセス/サーバスレッド	ProxyTimeout
3	Java EE サーバリクエストタイムアウト	アプリケーションの実行時間	Java EE サーバ 実行スレッド	configs.config.*構成名*.network-config.protocols.protocol.*構成名*.http.request-timeout-seconds
4	トランザクションタイムアウト	トランザクション処理時間	トランザクションサービス	configs.config.*構成名*.transaction-service.timeout-in-seconds
5	JAX-WS タイムアウト	Web サービス実行時間	JAX-WS、および WebService	configs.config.*構成名*.hitachi-jaxws-config.request-timeout
6	JAX-RS タイムアウト	Web サービス実行時間	JAX-RS、および Resource	configs.config.*構成名*.hitachi-jaxrs-config.read-timeout
7	JDBC SQL タイムアウト	SQL の実行時間	JDBC 接続プール	次のどれかの方法で指定します。 • create-jdbc-connection-pool コマンドの--statementtimeout オプション • glassfish-resources.xml の/resources/jdbc-connection-pool の statement-timeout-in-seconds 属性

重要

トランザクションタイムアウト発生時の対処

トランザクションタイムアウトが発生したとき、システム運用者は、ログに出力されたメッセージ（KDKD20033-W）を調査し、タイムアウトの原因を排除する必要があります。

なお、メッセージ（KDKD20033-W）は、ローカルトランザクションでタイムアウトが発生した場合だけ出力されます。

接続タイムアウト

接続タイムアウトの制御内容、設定するパラメーターを次の表に示します。

項番	設定項目名	制御内容	制御対象	設定するパラメーター
1	JAX-WS 接続タイムアウト	Web サービスに対する接続を確立するまでの時間	JAX-WS、および WebService	configs.config.*構成名*.hitachi-jaxws-config.connect-timeout
2	JAX-RS 接続タイムアウト	Web サービスに対する接続を確立するまでの時間	JAX-RS、および Resource	configs.config.*構成名*.hitachi-jaxrs-config.connect-timeout
3	JDBC 接続取得タイムアウト	JDBC コネクションプールから、 コネクションを取得する最大の待ち時間	JDBC コネクションプール	次のどれかの方法で指定します。 • create-jdbc-connection-pool コマンドの--maxwait オプション • glassfish-resources.xml の/resources/jdbc-connection-pool のmax-wait-time-in-millis 属性

Java EE サーバのサブコマンドのタイムアウト

asadmin ユーティリティーコマンドのサブコマンドの処理は、asadmin ユーティリティーコマンドのプロセスやドメイン管理サーバのプロセスなどのプロセス間で HTTP 通信をすることで実現されます。プロセス間の通信やasadmin ユーティリティーコマンドのサブコマンド自体の処理などに一定以上の時間を要した場合に、タイムアウトが発生します。asadmin ユーティリティーコマンドのサブコマンドには、ローカルサブコマンドとリモートサブコマンドがあります。それぞれの場合のタイムアウトについて説明します。

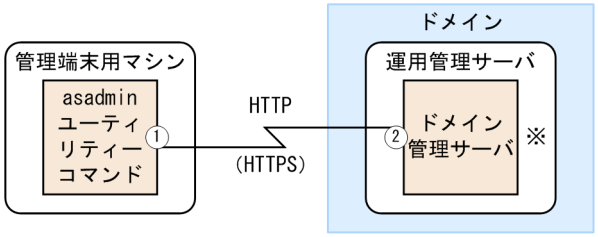
ローカルサブコマンドの場合

ローカルサブコマンドの場合は、asadmin ユーティリティーコマンドを実行したホスト上でサブコマンドが処理されるため、通信に関するタイムアウトは発生しません。ただし、サブコマンドによっては、コマンド処理タイムアウトを設定できるため、設定した値を超えるとタイムアウトが発生します。

リモートサブコマンドの場合

リモートサブコマンドの場合は、ドメイン管理サーバ上でサブコマンドが処理されるため、通信が発生し、タイムアウトが発生することがあります。 リモートサブコマンドを実行した場合のタイムアウトの発生個所を次の図に示します。

リモートサブコマンドのタイムアウト発生個所（サーバインスタンスの実行なしの場合）



（凡例）

：プロセス

①：asadminユーティリティーコマンドリードタイムアウトが発生する個所

②：ドメイン管理サーバリクエストタイムアウトが発生する個所

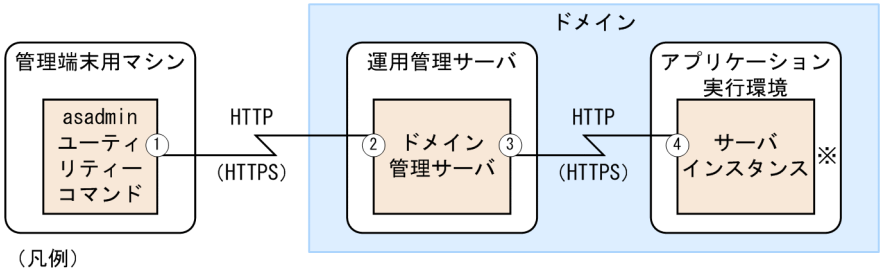
注※：asadminユーティリティーコマンドで個別にコマンド処理タイムアウトを設定している場合は、この個所でタイムアウトが発生します。

リモートサブコマンドを実行したとき、asadmin ユーティリティーコマンドのプロセスはサブコマンドの処理をドメイン管理サーバに委譲します。このとき、次の処理に対してタイムアウトが設定されます。

asadmin ユーティリティーコマンドのプロセスからドメイン管理サーバへの通信に対して設定する HTTP レスポンス読み取り処理

asadmin ユーティリティーコマンドのプロセスから受け取ったサブコマンドを含む HTTP リクエストの処理

リモートサブコマンドのタイムアウト発生個所（サーバインスタンスの実行ありの場合）



（凡例）

：プロセス

①：asadminユーティリティーコマンドリードタイムアウトが発生する個所

②：ドメイン管理サーバリクエストタイムアウトが発生する個所

③：ドメイン管理サーバリードタイムアウトが発生する個所

④：サーバインスタンスリクエストタイムアウトが発生する個所

注※：asadminユーティリティーコマンドで個別にコマンド処理タイムアウトを設定している場合は、この個所でタイムアウトが発生します。

リモートサブコマンドを実行したとき、asadmin ユーティリティーコマンドのプロセスはサブコマンドの処理をドメイン管理サーバに委譲します。ドメイン管理サーバはasadmin ユーティリティーコマンドのプロセスから受け取ったサブコマンドを処理した後に、サブコマンドやユーザーが指定したオプションによっては、サーバインスタンス上でも同様のサブコマンドを実行します。このとき、次の処理に対してタイムアウトが設定されます。

asadmin ユーティリティーコマンドのプロセスからドメイン管理サーバへの通信に対して設定する HTTP レスポンス読み取り処理

asadmin ユーティリティーコマンドのプロセスから受け取ったサブコマンドを含む HTTP リクエストの処理

ドメイン管理サーバプロセスからサーバインスタンスへの通信に対して設定する HTTP レスポンス読み取り処理

ドメイン管理サーバから受け取ったサブコマンドを含む HTTP リクエストの処理

サブコマンドのタイムアウトの詳細を次の表に示します

表 3-1　サブコマンドのタイムアウト一覧

項番	名称	発生プロセス	対象処理	タイムアウト時の動作	設定方法
1	asadmin ユーティリティーコマンドリードタイムアウト	asadmin ユーティリティーコマンドのプロセス	asdmin ユーティリティーコマンドのプロセスからドメイン管理サーバへの通信に対して設定する HTTP レスポンス読み取り処理	• asadmin ユーティリティーコマンドの実行に失敗しますが、サブコマンドの処理はドメイン管理サーバで継続します。ドメイン管理サーバでのサブコマンドの実行結果は、ドメイン管理サーバのホストのメッセージログにメッセージ（成功時：KDKD10162-I、 失敗時：KDKD10163-E）が出力されます。※1 • asadmin ユーティリティーコマンドのプロセスの標準出力には、タイムアウトを示すメッセージが出力されます。※2 • asadmin ユーティリティーコマンドのメッセージログには、エラーメッセージ（KDKD10164-E）が出力されます。	asadmin ユーティリティーコマンドの実行ホストの環境変数 AS_ADMIN_READTIMEOUT にタイムアウト値を設定します（単位：ミリ秒）。※3
2	ドメイン管理サーバリクエストタイムアウト	ドメイン管理サーバプロセス	asadmin ユーティリティーコマンドから受け取ったサブコマンドを含む HTTP リクエストの処理	• asadmin ユーティリティーコマンドの実行に失敗しますが、サブコマンドの処理自体はエラーにならないで継続する場合があります。※4 • asadmin ユーティリティーコマンドのプロセスの標準出力には、サブコマンドの実行エラーを示すメッセージが出力されます。※5 • ドメイン管理サーバのメッセージログには、エラーメッセージ	set parameter (configs.config.server-config.network-config.protocols.protocol.*リスナー名*.http.request-timeout-seconds)で設定します（単位：秒）。 *リスナー名*: admin-listener（secure admin が無効な場合） sec-admin-listener（secure admin が有効な場合）

項番	名称	発生プロセス	対象処理	タイムアウト時の動作	設定方法
				（KDKD10165-E）が出力されます。	
3	ドメイン管理サーバリードタイムアウト	ドメイン管理サーバプロセス	ドメイン管理サーバプロセスからサーバインスタンスへの通信に対して設定するHTTP レスポンス読み取り処理	• asadmin ユーティリティーコマンドの実行に失敗しますが、サーバインスタンス上でのサブコマンドの処理は継続します。サーバインスタンス上でのサブコマンドの実行結果は、サーバインスタンスのメッセージログにメッセージ（成功時：KDKD10162-I、 失敗時：KDKD10163-E）が出力されます。※1 • asadmin ユーティリティーコマンドのプロセスの標準出力には、サブコマンドの実行エラーを示すメッセージが出力されます。※5 • ドメイン管理サーバのメッセージログには、エラーメッセージ（KDKD10164-E）が出力されます。	ドメイン管理サーバのホストの環境変数 AS_ADMIN_READTIMEOUT にタイムアウト値を設定します（単位：ミリ秒）。※3
4	サーバインスタンスリクエストタイムアウト	サーバインスタンスプロセス	ドメイン管理サーバから受け取ったサブコマンドを含むHTTP リクエストの処理	• asadmin ユーティリティーコマンドの実行に失敗しますが、コマンド処理自体はエラーにならないで継続する場合があります。※4 • asadmin ユーティリティーコマンドのプロセスの標準出力には、コマンドの実行エラーを示すメッセージが出力されます。※5 • ドメイン管理サーバのメッセージログには、エラーメッセージ（KDKD10165-E）が出力されます。	set parameter (configs.config.*Java EEサーバのコンフィグ名*.network-config.protocols.protocol.sec-admin-listener.http.request-timeout-seconds)で設定します（単位：秒）。

注※1

実行結果ログを出力するサブコマンドと出力しないサブコマンドがあります。実行結果ログの出力の有無については、処理に時間が掛かるサブコマンドの表を参照してください。

注※2

次のメッセージが出力されます。

```
No response from Domain Admin Server after タイムアウト値 seconds.
The command is either taking too long to complete or the server has failed.
Please see the server log files for command status.
Command コマンド名 failed.
```

注※3

asadmin ユーティリティーコマンドリードタイムアウト、ドメイン管理サーバリードタイムアウトの設定値は各サブコマンドで共通の値になります。デフォルトのタイムアウト時間でタイムアウトが発生する場合は、最も処理に時間が掛かるコマンドの時間に合わせて、それらのタイムアウト時間を設定する必要があります。

注※4

タイムアウト時に処理を継続するサブコマンドについては、処理に時間が掛かるサブコマンドの表を参照してください。

注※5

次のメッセージが出力されます。

```
Command コマンド名 failed.
```

処理に時間が掛かるサブコマンドを次の表に示します。

表 3-2　処理に時間が掛かるサブコマンド

項番	コマンド名	条件				実行結果ログ
		デプロイするアプリケーション数	デプロイするアプリケーションのサイズ	リクエスト終了待ち時間	収集対象のファイルサイズおよびファイル数	
1	start-instance	○	－	－	－	出力する
2	stop-instance	○	－	○	－	出力する
3	start-cluster	○	－	－	－	出力する
4	stop-cluster	○	－	－	－	出力する
5	start-servers	○	－	－	－	出力しない
6	stop-servers	○	－	○	－	出力しない
7	deploy	－	○	－	－	出力する
8	undeploy	－	○	○	－	出力する
9	enable	－	○	－	－	出力する
10	disable	－	－	○	－	出力する
11	collect-snapshot	－	－	－	○	出力しない

（凡例）

○：条件によってサブコマンドの処理時間が変動します。

－：サブコマンドの処理時間は変動しません。

次のどれかが発生した場合、それらのタイムアウト時間にコマンド処理タイムアウトより十分に大きな時間を設定し、それらのタイムアウトが発生しないようにしてください。

- asadmin ユーティリティーコマンドリードタイムアウト
- ドメイン管理サーバリクエストタイムアウト
- ドメイン管理サーバリードタイムアウト
- サーバインスタンスリクエストタイムアウト

タイムアウト値を設定する場合、これらのタイムアウト値には、すべて同じタイムアウト時間を設定してください。

タイムアウト時間は、asadmin ユーティリティーコマンドのサブコマンドごとではなく、すべてのサブコマンドで同じ値が適用されます。そのため、最も処理時間が掛かるサブコマンドの時間に合わせてタイムアウト時間を設定してください。

メモ

処理に時間が掛かるサブコマンドは、デプロイするアプリケーション数に依存して時間が掛かるなど、条件によってコマンドの処理時間が増加し、タイムアウトになることがあります。

3.8　セキュリティー対策について

Application Server を使用したシステムのセキュリティー対策とは、暗号化やユーザー認証などの機能で、データの改ざんやハッキングなどの脅威からデータやシステムを守ることです。Application Server の機能としてのセキュリティー対策のほか、システムの運用方法を規定して脅威からデータやシステムを守るセキュリティー対策も必要です。

セキュリティー対策の方針

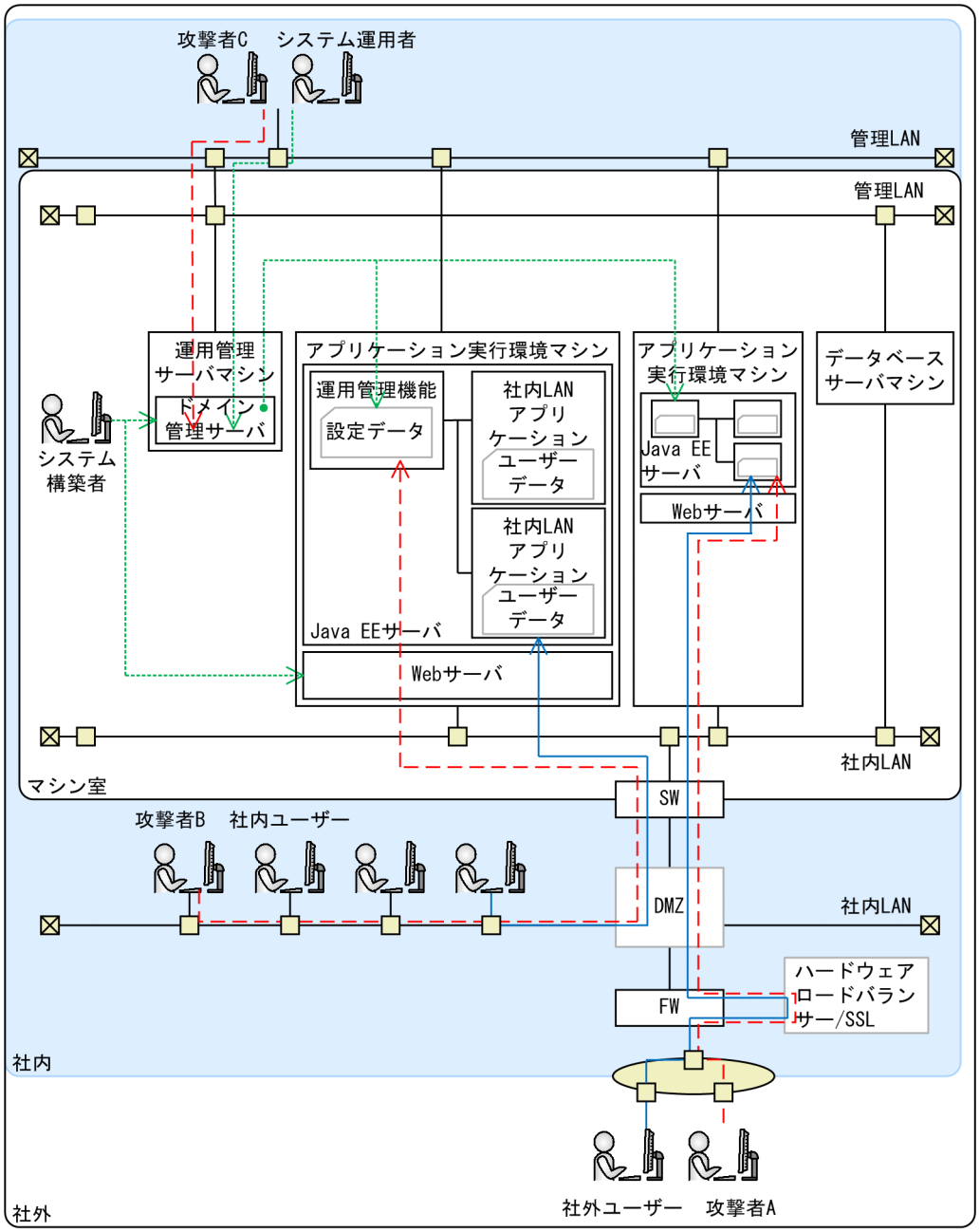
Application Server を利用した標準的なシステム構成を基に、そこで想定される脅威を挙げ、その脅威への対応方針と対策を示します。 なお、Application Server 上で動作するアプリケーションのセキュリティーについては、アプリケーションの開発元が考慮する内容のため、ここでは説明の対象外とします。

Application Server を利用した標準的なシステム構成と想定される脅威

Application Server を利用した標準的なシステム構成を例に、運用シナリオと想定される脅威を次に示します。

Application Server を利用した標準的なシステム構成

Application Server を利用した標準的なシステム構成を次の図に示します。



システムの構成要素

Application Server を利用した標準的なシステムの構成要素を次の表に示します。

表 3-3　システムの構成要素

項番	要素	説明
1	社内	社内ユーザーおよびシステム運用者が作業するエリアです。このエリアは、社内 LAN および管理 LAN が敷設されていて、管理 LAN と社内

項番	要素	説明
		LAN は物理的に分離されています。社内の管理 LAN と社内 LAN は、どちらも社外からは物理的に分離されていることを想定しています。
2	マシン室	システム構築者 が作業するエリアです。セキュリティーを確保するため、運用管理サーバマシン、アプリケーション実行環境マシン、データベースサーバマシンなどを配置しています。 このエリアは、入退室管理などが厳格に行われているため、社内で一番セキュリティーレベルが高いことを想定しています。
3	社内 LAN	社内ユーザーが使う PC とアプリケーションとの間で、業務用データをやりとりするための LAN です。
4	管理 LAN	システム運用者が使う PC とドメイン管理サーバとの間で、管理用データをやりとりするための LAN です。
5	DMZ	社内 LAN と WWW の間に設置された隔離されたネットワーク領域です。
6	ドメイン管理サーバ	Application Server がインストールされた複数のアプリケーション実行環境マシンを、ドメイン単位にグループ化して運用を管理するサーバです。
7	運用管理サーバマシン	ドメイン管理サーバがインストールされているマシンです。アプリケーション実行環境マシンの運用を管理します。
8	アプリケーション実行環境マシン	Application Server がインストールされているマシンです。アプリケーションを管理します。
9	データベースサーバマシン	DBMS が動作するサーバです。アプリケーションが使用することを想定しています。 なお、データベースのセキュリティー機能は、Application Server の管理対象外です。
10	スイッチ	IP アドレスによるルーティングの制御をします。
11	ファイアウォール	IP アドレスとポート番号などによるアクセス制御ができます。
12	ハードウェアロードバランサー/SSL	アプリケーション実行環境マシンへのリクエストを分散させることができます。SSL 通信をサポートしているため、暗号化した通信もできます。

システムの運用シナリオ

Application Server を利用した標準的なシステム構成図を基にした運用シナリオを次に示します。

1. 社外ユーザーは、PC にインストールされた Web ブラウザーを使用して HTTP や WebSocket でアプリケーションにアクセスし、インターネット経由でシステムを利用します。

   なお、社外ユーザーは、Web サービス（JAX-WS、JAX-RS）や RMI/RMI-IIOP を使用したアクセスはしないと想定します。

2. 社内ユーザーは、社内から業務用の PC にインストールされた Web ブラウザーまたはアプリケーションのクライアントを利用してアプリケーションに HTTP（SOAP/REST 含む）や WebSocket でアクセスし、社内 LAN 経由で業務を行います。

   なお、社内ユーザーは、RMI/RMI-IIOP を使用したアクセスはしないと想定します。

3. システム運用者は、社内から管理用の PC にインストールされた Web ブラウザーまたは Application Server の管理コマンドを使用してドメイン管理サーバ にアクセスし、管理 LAN 経由で運用時の管理をします。
4. システム構築者 は、運用管理サーバマシン、アプリケーション実行環境マシンおよびデータベースサーバマシンに直接アクセスし、セットアップおよび設定変更をします。
5. システム構築者 は、マシン室から、管理用の PC にインストールされた Web ブラウザーまたは Application Server の管理コマンドを使用してドメイン管理サーバ にアクセスし、複数のアプリケーション実行環境マシンの管理をします。
6. ドメイン管理サーバ はシステム運用者あるいはシステム構築者 から受けた指示を各アプリケーション実行環境マシンに送信し、アプリケーション実行環境マシン上の Application Server プロセスの起動、停止、および設定変更をします。

想定されるシステムへの攻撃

Application Server を利用したシステムに対する攻撃者による脅威を次に示します。

1. 攻撃者 A は、インターネット上からアプリケーション実行環境マシンを直接攻撃します。
2. 攻撃者 B は、社内 LAN に接続している業務用の PC から、不正にアプリケーションへアクセスし、攻撃します。
3. 攻撃者 C は、システム運用者になりすまして、管理 LAN に接続している業務用 PC から不正にシステムへアクセスし、ドメイン管理サーバやアプリケーション実行環境マシンを攻撃します。

システム上で想定される脅威に対するセキュリティーポリシー

Application Server を利用したシステムのセキュリティーポリシーの想定を示します。

ネットワークのセキュリティーポリシーの想定

1. レイヤー 3 までのネットワークに関するセキュリティーでは、次の内容を考慮して設計します。

   社外からのパケットは、ハードウェアロードバランサー/ファイアウォールによって通常使用されるポートに限定したアクセスに制限する。

   社内 LAN および社外からのパケットは管理用の LAN に送信されないようにする。
2. レイヤー 4 以上のネットワークに関するセキュリティーでは、次の内容を考慮して設計します。

   Java EE の標準仕様の範囲内で対応し、そのほかについてはアプリケーション開発者の責任にする。
3. 管理 LAN のネットワークに関するセキュリティーでは、次の内容を考慮して設計します。

   管理 LAN は社内 LAN とは独立しているが、悪意の有無に関係なく、攻撃者に接続させないようにする。

物理的操作のセキュリティーポリシーの想定

1. 社外は、次の内容を考慮して設計します。

   悪意のある人物がいる可能性があるため、社外からは、社内 LAN および管理 LAN に物理的に接続させないようにする。
2. 社内は、次の内容を考慮して設計します。

悪意のある攻撃者がいるため、Application Server への物理的アクセスはネットワーク経由に限定する。

3. マシン室エリアは、次の内容を考慮して設計します。

   悪意の有無に関係なく、攻撃者がエリアに入れないようにする。

開発環境のセキュリティーポリシーの想定

1. 社内 LAN 上のマシンでアプリケーションを開発します。
2. 開発者が悪意のあるコードをアプリケーションに混入させる恐れがないか確認します。
3. 開発したアプリケーションに対して、有識者がレビューを実施します。

   開発したアプリケーションがウイルスに感染していないことを確認する。
4. 社内 LAN 上で開発したアプリケーションは、システム構築者が管理 LAN を経由してアプリケーション実行環境マシンにデプロイします。

   管理 LAN 上のアプリケーション実行環境マシンに、アプリケーション開発者がアプリケーションを直接デプロイさせない。

Application Server のセキュリティーポリシー

想定する脅威に対する Application Server のセキュリティーポリシーを次に示します。

1. 社内 LAN からのアプリケーションに対するアクセスでは、システム構築者が指定した権限の範囲内で運用します。

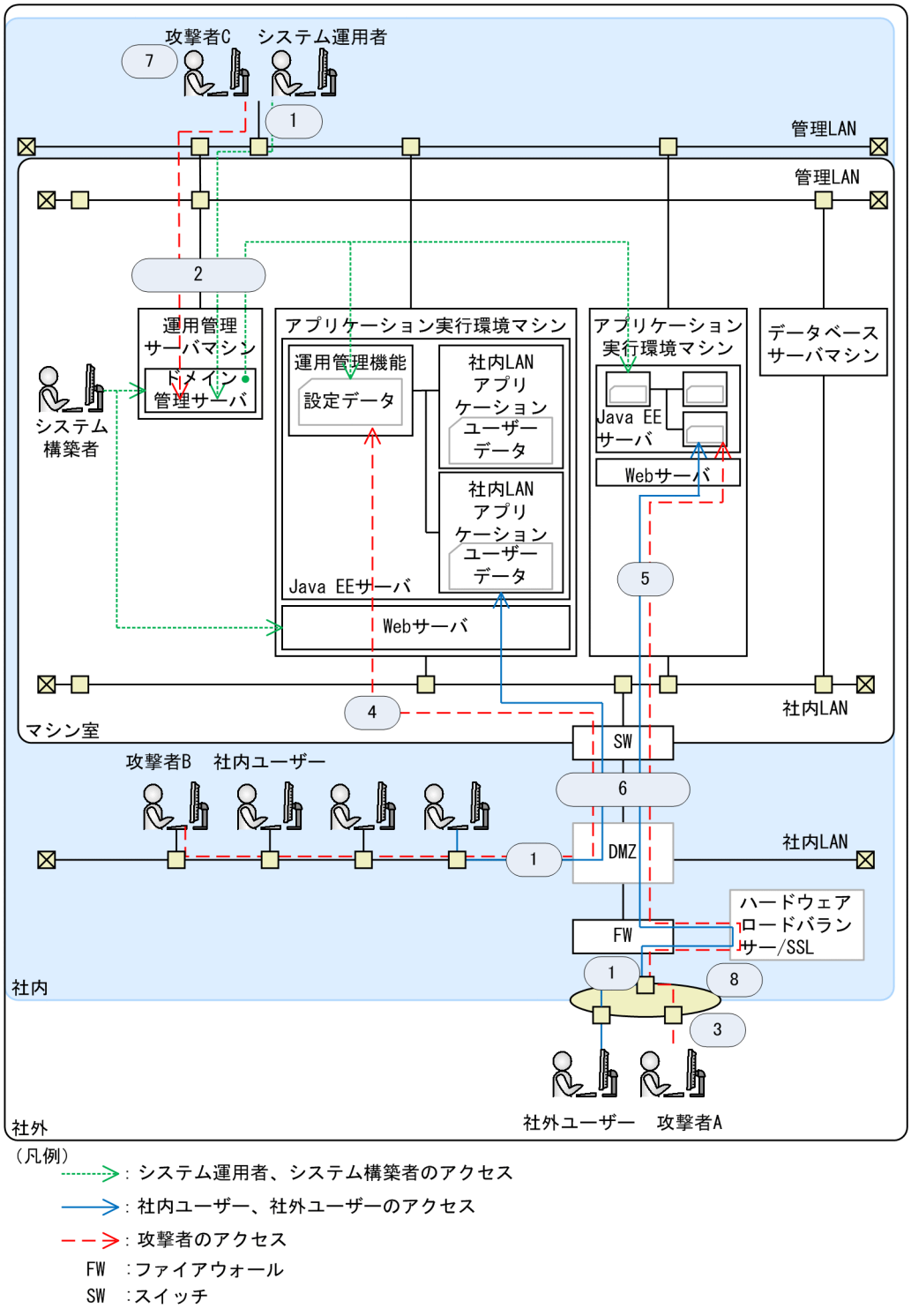
   社内 LAN からのアプリケーションに対するアクセスでは、システム構築者が指定した権限を逸脱しないようにします。

   指定したファイルアクセス権限を逸脱しないようにします。
2. 社内 LAN からのアプリケーションに対するアクセスは盗聴・改ざん防止のための暗号化をサポートします。
3. 社内 LAN からのアプリケーションに対するアクセスは、アプリケーションのログと Application Server のログをそれぞれ取得します。
4. 社内 LAN からのアプリケーションに対するアクセスは認証機能を Java EE の標準仕様の範囲内で提供します。
5. 運用管理機能に使用されるポートのセキュリティーは保証しないようにします。
6. Application Server のファイルおよびメモリの耐タンパ性は保証しないようにします。
7. Application Server のファイルおよびメモリで保護すべきデータの出力は制限しないようにします。
8. Application Server からのネットワークを超えた出力については、セキュリティーに関連した設定データ（Java EE サーバの設定ファイルの domain.xml、パスワード管理ファイルなど）は保護します。

システムに対する脅威と対策

Application Server を利用した標準的なシステム構成図を基に、システムで発生するおそれがあるセキュリティー上の脅威について説明します。

図中の丸付き数字は脅威が想定される個所を示しています。



（凡例）

：システム運用者、システム構築者のアクセス

：社内ユーザー、社外ユーザーのアクセス

：攻撃者のアクセス

FW　：ファイアウォール

SW　：スイッチ

システムに対する脅威への対策を次の表に示します。なお、図中の丸付き数字と表の項番が対応しています。

項番	システムに対する脅威	脅威の概要	Application Server の機能、および運用面での対策	セキュリティー機能の設定
1	通信データののぞき見、および改ざん	ネットワークに流れるパケットを不正に取得して業務データや管理データを不正に閲覧したり、パケット自体を改変してデータを不正に変更したりすることです。	機能 Web Server の SSL、および TLS 暗号機能を利用します。 運用 ロードバランサーの SSL、および TLS 暗号機能を利用します。	Web サーバの動作環境を設定するファイル（httpsd.conf）で、SSL 認証の暗号機能を指定します。
2	指令の改ざん	ネットワークに流れるパケットを改変して、システム構築者や運用者が発行した指令を改変することです。	運用 管理 LAN のアクセス制御を実施します。	--
3	権限がないユーザーのアプリケーションへのアクセス	攻撃者が、アクセス権限のないアプリケーションにリクエストを送信し、不正に業務を行うことです。	機能 Java EE 標準のセキュリティーロールを使用します。	create-auth-realm サブコマンドで、名前付き認証レルムを作成し、アプリケーションの DD でセキュリティーロールを定義します。
4	権限がないユーザーの指令の実行	アクセス権がない攻撃者が、運用管理機能にアクセスし、不正に指令を実行することです。	機能 ドメイン管理サーバの管理パスワードで認証します。 運用 ドメイン管理サーバのユーザー認証用のデータ（パスワードなど）は、管理 LAN だけに接続された機器だけに保存します。 ドメイン管理サーバの管理ポートは、アクセス元を管理 LAN 側だけに限定します。	create-domain サブコマンドでドメインを作成する際に、管理ユーザーのパスワードを指定します。管理ユーザーのパスワードは、change-admin-password サブコマンドで変更できます。
5	アプリケーションの脆弱性が連鎖的に Java EE サーバ全体に及ぼす影響	特定のアプリケーションが受信した想定外の電文によって脆弱性が発現し、それが連鎖的に Java EE サーバ全体へ影響することです。	機能 Java のセキュリティーポリシーによって保護（API 利用の制限）します。	Java SE が提供するシステムプロパティー（server.policy ファイル）のロケーションを指定します 。
6	不正アクセスの否認	攻撃者が不正アクセスをしたにもかかわらず、攻撃者が不正アクセスの事実を否認することです。	機能 Web Server のアクセスログを設定します。	Web サーバの動作環境を設定するファイル（httpsd.conf）で、Web サーバのアクセスログを設定します。
7	ショルダーハッキング	攻撃者がシステム構築者やシステム運用者の肩越しにパスワードなどの情報を盗み見ることです。	機能 パスワードなどの重要な情報を見られないにするため、テキストボックスに入力した文字を隠して、代わりに黒丸や別の文字を表示する機能を使用します。	--

項番	システムに対する脅威	脅威の概要	Application Server の機能、および運用面での対策	セキュリティー機能の設定
8	サービス拒否攻撃	攻撃をしてサービスの提供ができない状態にすることです。	機能 タイムアウト制御やパケットサイズの制限をするための機能を使用します。 運用 JP1 と連携して、セキュリティー異常監視をします。 その他 ファイアウォール、ロードバランサーなどのネットワーク製品で対策をして、DoS 攻撃のアクセスを軽減する必要があります。	Web サーバの動作環境を設定するファイル（httpsd.conf）で、タイムアウト制御やパケットサイズの制限を指定します。

関連項目

- 6.2.1　リバースプロキシーを設定する
- 6.2.2　SSL を設定する

3.9 Application Server で利用できるアプリケーション認証について

アプリケーション認証とは、セキュリティーを確保するために使用するユーザー認証のことです。Application Server で利用できるアプリケーション認証の方式には、クライアント認証、BASIC 認証、FORM 認証、および DIGEST 認証があります。

アプリケーション認証の分類

Application Server で利用できるアプリケーション認証について次の表に示します。認証を行う個所が複数ある認証方式については、どれか 1 個所を設定します。

項番	認証方式	認証を行う個所	認証の仕組み
1	クライアント認証	Web Server	SSL
2		SSL アクセラレーター	
3	BASIC 認証	Web Server	パスワードファイル
4			LDAP リポジトリー
5		Java EE サーバ	JAAS
6	FORM 認証	Java EE サーバ	JAAS
7	DIGEST 認証	Java EE サーバ	

3.10　リソースの見積もりについて

リソースは、共用メモリーサイズ、プロセス数、スレッド数、ファイルディスクリプター数、およびデータベース容量を見積もったあと、それらを集計して全体のリソースを見積もります。

共用メモリーサイズの見積もり

Web サーバ、およびパフォーマンストレーサーが使用する共用メモリーサイズの見積もりを次に示します。

Web サーバが使用する共用メモリーサイズ

Web サーバが使用する共用メモリーサイズは、次の計算式で算出します。

なお、Web サーバが使用する共用メモリーサイズは共用メモリーセグメントのサイズです。

Webサーバが使用する共用メモリーサイズ＝409,600＋7,168×MaxRequestWorkers

（凡例）

MaxRequestWorkers：同時に接続できるクライアントの最大数（バイト）

パフォーマンストレーサーが使用する共用メモリーサイズ

パフォーマンストレーサーが使用する共用メモリーサイズは、次の計算式で算出します。

なお、パフォーマンストレーサーが使用する共用メモリーサイズは共用メモリーセグメントのサイズです。

パフォーマンストレーサーが使用する共用メモリーサイズ＝*PrfTraceBufferSize指定値*×1,024＋18,496

（凡例）

PrfTraceBufferSize指定値：共用メモリーに確保するパフォーマンストレーサーのトレースバッファーサイズ（バイト）

プロセス数の見積もり

Java EE Server、Web サーバ、およびパフォーマンストレーサーのプロセス数の見積もりを次に示します。

Java EE Server のプロセス数

Java EE Server の常駐プロセス数は、次の計算式で算出します。

Java EE Serverの常駐プロセス数＝*ドメイン管理サーバ数*＋*サーバインスタンス数*

Web サーバのプロセス数

Web サーバのプロセス数は、次の計算式で算出します。

Webサーバのプロセス数＝*制御プロセス*＋*サーバプロセス*＋*CGIプロセス*＋*gcacheサーバ*＋*rotatelogsプロセス*＋*rotatelogs2プロセス*

（凡例）

制御プロセス：1

サーバプロセス：MaxRequestWorkers ディレクティブ指定数

CGIプロセス：サーバプロセス数分

gcacheサーバ：1

rotatelogsプロセス：CustomLog ディレクティブ、ErrorLog ディレクティブ、HWSRequestLog ディレクティブ、TransferLog ディレクティブに指定した rotatelogs プログラム数

rotatelogs2プロセス：CustomLog ディレクティブ、ErrorLog ディレクティブ、HWSRequestLog ディレクティブ、TransferLog ディレクティブに指定した rotatelogs2 プログラム数

パフォーマンストレーサーのプロセス数

パフォーマンストレーサーの常駐プロセス数は 1 です。

スレッド数の見積もり

Java EE Server、Web サーバ、およびパフォーマンストレーサーのスレッド数の見積もりを次に示します。

Java EE Server のスレッド数

Java EE Server が使用するスレッド数は、次の計算式で算出します。

$$\text{Java EE Serverが使用するスレッド数} = \sum_{i=1}^{m} Dt(i) + \sum_{i=1}^{n} SIt(i)$$

（凡例）

Dt：ドメイン管理サーバの 1 プロセス当たりのスレッド総数

SIt： サーバインスタンスの 1 プロセス当たりのスレッド総数

m：ドメイン管理サーバのプロセス数

n：サーバインスタンスのプロセス数

ドメイン管理サーバの 1 プロセス当たりのスレッド総数Dt と、サーバインスタンスの 1 プロセス当たりのスレッド総数SIt を求める計算式を次に示します。

$$\text{ドメイン管理サーバの1プロセス当たりのスレッド総数} = \sum_{i=1}^{m} Tm(i) + Tn$$

$$\text{サーバインスタンスの1プロセス当たりのスレッド総数} = \sum_{i=1}^{n} Tm(i) + \sum_{i=1}^{r} (Cm(i) \times 2) + Tn$$

（凡例）

Tm：スレッドプールの最大値

Cm：JDBC コネクションプールのコネクション最大値（コネクション障害検知機能が有効な場合だけ。コネクション障害検知機能が無効な場合は 0。）

Tn：スレッドプールに属さないスレッド数。ドメイン管理サーバやサーバインスタンスがデフォルトで生成するスレッド数は 44 です。

m：ドメイン管理サーバのスレッドプール数

n：サーバインスタンスのスレッドプール数

r：サーバインスタンスの JDBC コネクションプール数

mとnのスレッドプール数は、list-threadpoolsコマンドを実行して表示されるスレッドプールの一覧で確認してください。

Webサーバのスレッド数

Webサーバが使用するスレッド数は、Webサーバが使用するプロセス数と同じです。

パフォーマンストレーサーのスレッド数

パフォーマンストレーサーが使用するスレッド数は10です。

ファイルディスクリプター数の見積もり

Java EEサーバ、Webサーバ、およびパフォーマンストレーサーのファイルディスクリプター数の見積もりを次に示します。

Java EEサーバのファイルディスクリプター数

Java EEサーバのファイルディスクリプター数は、OSのopen()、socket()などのシステムコールの発行に従って増減します。また、ファイルディスクリプター数は、DBMSの同時接続数によっても異なります。

ここでは、DBMSの最大同時接続数を基に、Java EEサーバが使用するファイルディスクリプター数の見積もりを示します。

Java EEサーバが使用するファイルディスクリプター数は、次の計算式で算出します。

```
Java EEサーバのファイルディスクリプター数＝JDBコネクションプールサイズ×2
```

Webサーバのファイルディスクリプター数

Webサーバが使用するファイルディスクリプター数は、次の計算式で算出します。

- Linuxの場合

```
Webサーバが使用するファイルディスクリプター数＝(50＋A×B＋C＋11×C×D＋8×E＋(F＋H)×G)×1.2
```

- AIXの場合

```
Webサーバが使用するファイルディスクリプター数＝(50＋A×B＋C＋3×C×D＋5×E＋(F＋H)×G)×1.2
```

(凡例)

A：Listenディレクティブ指定数（Listenディレクティブの指定がない場合は1）

B：ホストに割り当てられたIPアドレスの数

C：CustomLogディレクティブ、ErrorLogディレクティブ、HWSRequestLogディレクティブ、TransferLogディレクティブ指定の総数

D：rotatelogsプログラムまたはrotatelogs2プログラムを使用する場合は1、使用しない場合は0

E：同時実行CGI数（MaxRequestWorkers指定値。CGIプログラムを使用しない場合は0）

F：SSLを使用する場合は3、使用しない場合は2

G：同時実行リクエスト数（MaxRequestWorkers指定値）

H：リバースプロキシーを使用する場合は 1、使用しない場合は 0

なお、CGI プログラム内および Web サーバに同梱されていない外部モジュール内で使用するファイルディスクリプターの数は含みません。

パフォーマンストレーサーのファイルディスクリプター数

パフォーマンストレーサー起動時に使用するファイルディスクリプターの数は、32 以上に設定してください。

データベース容量の見積もり

Application Server の機能として EJB タイマーサービスと Java Batch を使用する場合、DBMS を必要とします。DBMS にはテーブル情報が定義されている DDL ファイルが格納されています。この DDL ファイルのテーブル情報および実測したレコード数を基に、データベース容量の見積もりをしてください。

見積もりに必要な DDL ファイルの格納場所を次に示します。

表 3-4　DDL ファイルの格納場所

項番	対象	格納先ディレクトリー	ファイル名	DBMS
1	EJB タイマーサービス	*Application Server のインストールディレクトリー*/javaee/glassfish/lib/install/databases	ejbtimer_hirdb.sql	HiRDB
2			ejbtimer_oracle.sql	Oracle
3	Java Batch		jsr352-hirdb.sql	HiRDB

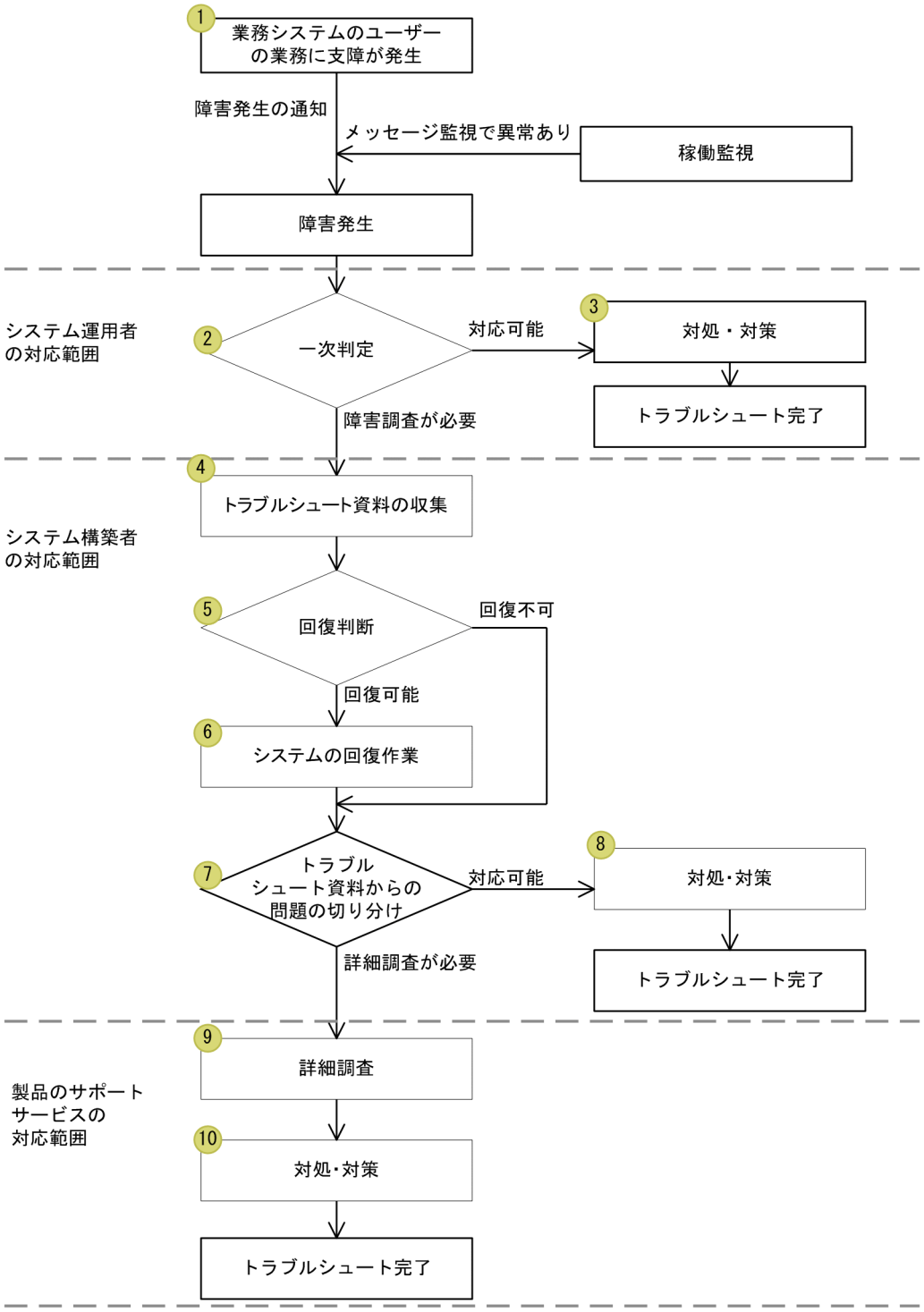
3.11　トラブルシュートの流れについて

Application Server のシステムで障害が発生した場合、トラブルシュートの流れに従って障害の調査や対処を実施します。システム運用時と、システム構築時とでは、トラブルシュートの流れが異なります。

システム運用時のトラブルシュートの流れ

システム運用時のトラブルシュートの流れについて説明します。

Application Server のシステム運用時に、障害が発生した場合のトラブルシュートの流れを次の図に示します。

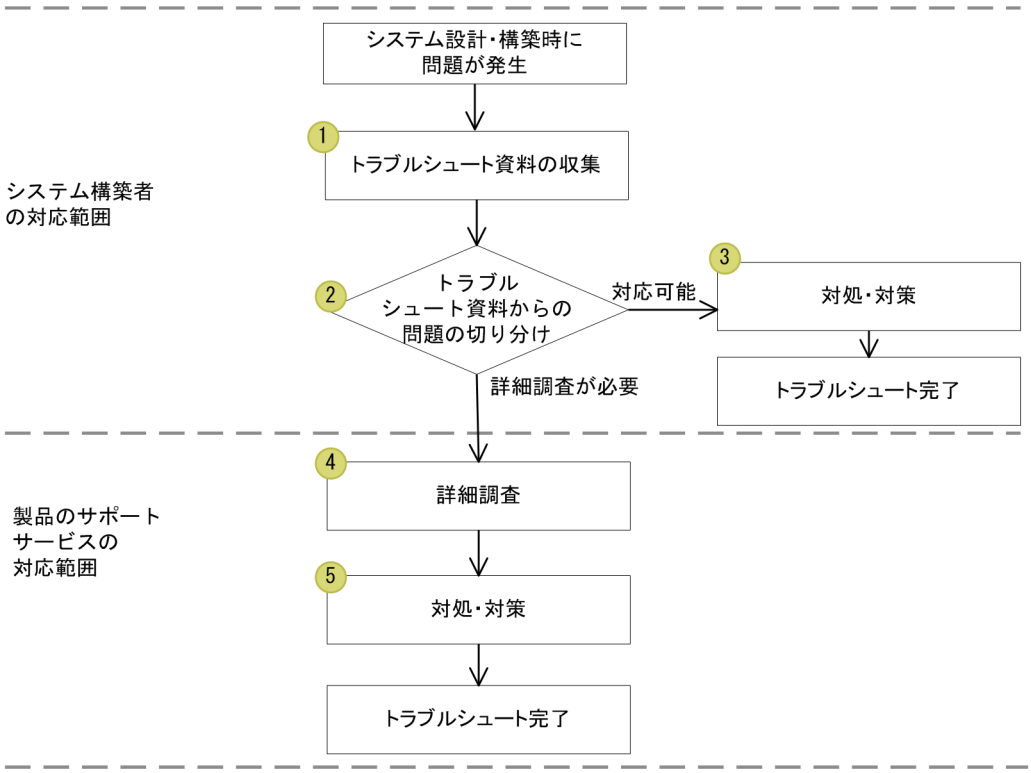


1. 業務システムのユーザーの業務に支障が発生した場合、ユーザーからシステム運用者に障害発生が通知されます。Application Server のメッセージ監視では、障害を知らせるメッセージが出力され、システム運用者に障害発生が通知されます。
2. 障害発生の通知を受け、システム運用者が障害の一次判定を実施します。
3. 一次判定の結果、対応できる場合は、障害の対処および対策を実施します。
4. 一次判定の結果、さらに障害調査が必要な場合は、システム構築者がトラブルシュート資料を収集します。

5. トラブルシュート資料を利用して、回復できるかどうかの回復判断を実施します。
6. 回復判断の結果、回復できる場合は障害を回復させます。
7. 回復判断の結果、回復できない場合、および回復可能で回復作業を実施した場合のどちらの場合も、トラブルシュート資料を利用して、問題を切り分けます。
8. 問題を切り分けた結果、システム構築者が対応できる場合は、障害の対処および対策を実施します。
9. さらに詳細な調査が必要な場合は、製品サポートサービスで詳細な調査を実施します。
10. 詳細な調査の結果、障害への対処および対策を実施します。

システム構築時のトラブルシュートの流れ

システム構築時に、障害が発生した場合のトラブルシュートの流れを次の図に示します。



1. 発生した障害について、システム構築者がトラブルシュート資料を収集します。
2. トラブルシュート資料を利用して、問題を切り分けます。
3. 問題を切り分けた結果、システム構築者が対応できる場合は、障害の対処および対策を実施します。
4. さらに詳細な調査が必要な場合は、製品サポートサービスで詳細な調査を実施します。
5. 詳細な調査の結果、障害への対処および対策を実施します。

関連項目

- 9.1　Application Server が出力するトラブルシュート資料について

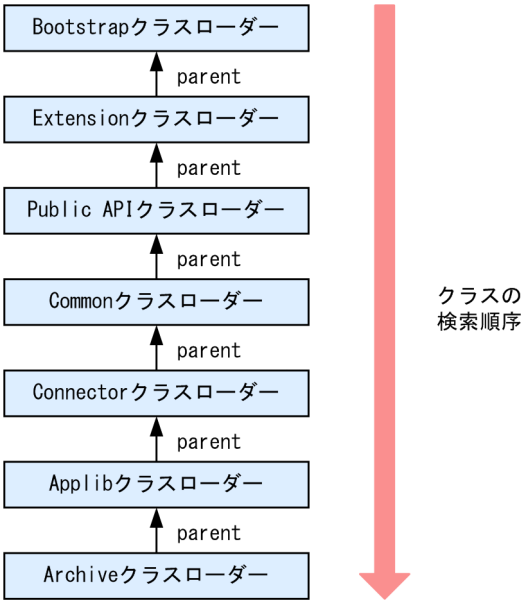
3.12　クラスローダーの構成について

クラスローダーとは、クラスをロードするオブジェクトです。アプリケーションで利用するライブラリーを開発するとき、ライブラリーのスコープを定義する必要があります。

クラスローダーの構成

アプリケーションで利用するライブラリーを開発するとき、ライブラリーのスコープを定義するには、Java EE Server のクラスローダーの階層構造をアプリケーション開発者が把握しておく必要があります。

Java EE Server のクラスローダーの階層構造を次に示します。



各クラスローダーの内容を次の表に示します。

項番	クラスローダー名	説明
1	Bootstrap	Java VM が提供するクラスをロードします。
2	Extension	system extensions ディレクトリー（domain-dir/lib/ext）に配置した JAR ファイルからクラスをロードします。
3	Public API	Java EE API および Application Server 固有 API とその実装クラスをロードします。
4	Common	アプリケーションやリソースアダプターで共通して使用するライブラリー（JAR ファイル）をロードします。 *Application Serverのインストールディレクトリー*/javaee/glassfish/lib ディレクトリーに配置された JAR ファイルからクラスをロードし、次に*Application Serverのインストールディレクトリー*/javaee/glassfish/domains/*ドメイン名*/lib からクラスをロードします。
5	Connector	すべてのアプリケーションが共用するリソースアダプターアーカイブをロードします。
6	Applib	デプロイ時に指定したライブラリー（JAR ファイル）をロードします。※

項番	クラスローダー名	説明
7	Archive	デプロイしたアプリケーションやモジュールに含まれる次のファイルおよびクラスをロードします。 • WAR ファイル、EAR ファイル、および JAR ファイル • サーバインスタンスが生成するアプリケーション固有のクラス（スタブクラスや JSP ページから作られるサーブレットなど）

注※

デプロイした複数のアプリケーションで同一ライブラリーを利用する場合、各アプリケーションは同じライブラリーのインスタンスを共有します。

ライブラリーからほかのライブラリーのクラスは参照できません。

4 アプリケーション実行環境の構築

アプリケーション実行環境を構築する手順について説明します。1 つの Java EE サーバを配置するシステム、または複数の Java EE サーバを配置しロードバランサーを利用するクラスター構成のシステムを構築します。システムの構築には、コマンドまたは Administration Console を利用します。なお、クラスター構成とは、複数の Java EE サーバをグループ化して管理するための構成のことです。

4.1　構築するアプリケーション実行環境について

アプリケーション実行環境は、1 つの Java EE サーバを配置する構成、または複数の Java EE サーバを配置して負荷を分散させるクラスター構成のどちらかの方法で構築できます。構築できるアプリケーション実行環境の構成や、構築方法の種別など、アプリケーション実行環境の構築の概要について説明します。

構築するアプリケーション実行環境の構成

このマニュアルで説明しているアプリケーション実行環境の構築手順では、次のどちらかの構成を構築できます。

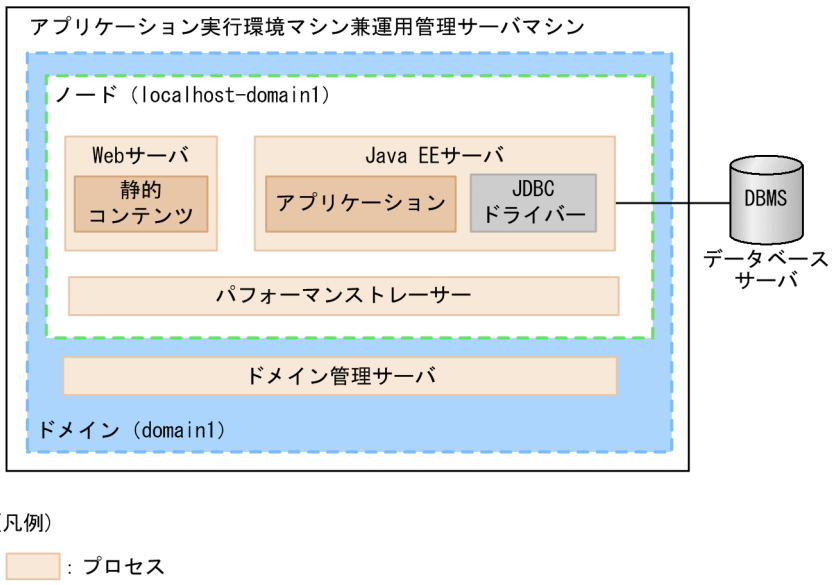
- 1 つの Java EE サーバを配置する構成

  Java EE サーバを 1 つ配置する構成です。次のマシンで構成されます。

  - アプリケーション実行環境マシン兼運用管理サーバマシン

    1 台のマシンに、1 つの Java EE サーバ、Web サーバ、パフォーマンストレーサー、およびドメイン管理サーバを配置します。各プロセスは、ドメインおよびノードで管理します。

  1 つの Java EE サーバを配置する構成の場合のアプリケーション実行環境の論理構成を次に示します。



- 複数の Java EE サーバを配置する構成

  クラスター構成にした複数の Java EE サーバを配置した構成です。別マシン上の複数の Java EE サーバをグループ化し、クラスター内の複数の Java EE サーバにリクエストを振り分けます。リクエストの振り分けにはロードバランサーを利用します。次のマシンで構成されます。

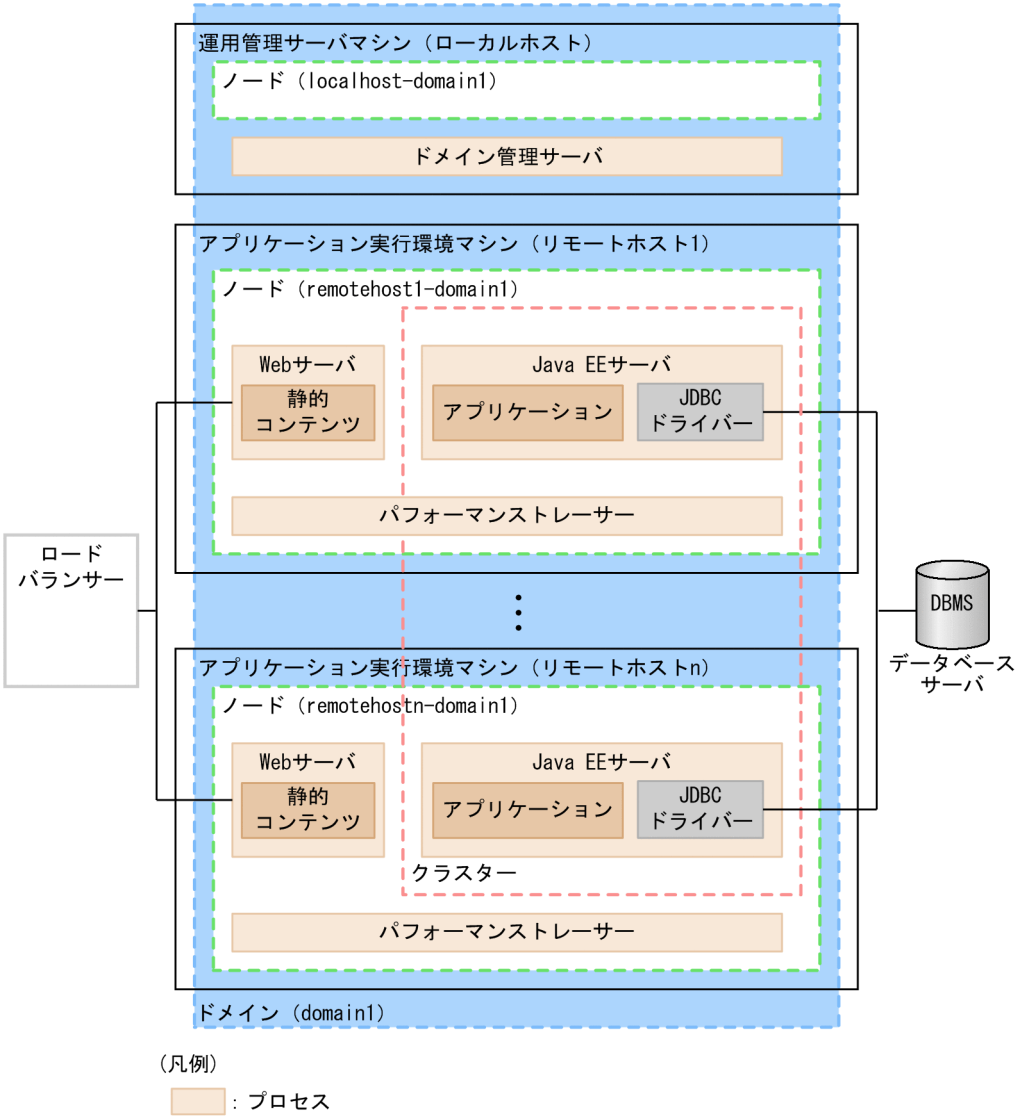
  - アプリケーション実行環境マシン

    複数台のアプリケーション実行環境マシンそれぞれに、Java EE サーバ、Web サーバ、およびパフォーマンストレーサーを配置します。各プロセスは、ドメイン、ノード、およびクラスターで管理します。

  - 運用管理サーバマシン

ドメイン管理サーバを配置します。

複数の Java EE サーバを配置する構成の場合のアプリケーション実行環境の論理構成を次に示します。



なお、構築するアプリケーション実行環境の構成によって構築手順や指定値などが異なる個所には、「1 つの Java EE サーバを配置する構成の場合」、または「複数の Java EE サーバを配置するクラスター構成の場合」と記載しています。

Application Server のインストールおよびセットアップ後にデフォルトで作成される要素

Application Server のインストールが完了すると、インストール先のホスト上に、次のコンポーネントが作成されます（[]内は名称を示します）。

- デフォルトのドメイン[domain1]
- デフォルトのノード[localhost-domain1]
- ドメイン管理サーバのスレッドプール[admin-thread-pool、 http-thread-pool、 thread-pool-1]

アプリケーション実行環境の構築方法の種別

アプリケーション実行環境は、次のどちらの方法でも構築できます。このマニュアルでは、コマンドを利用した方法で構築手順を説明しています。

- コマンド

  このマニュアルに記載されている手順のとおりに実行して構築します。コマンドは管理者権限で実行します。

- Administration Console

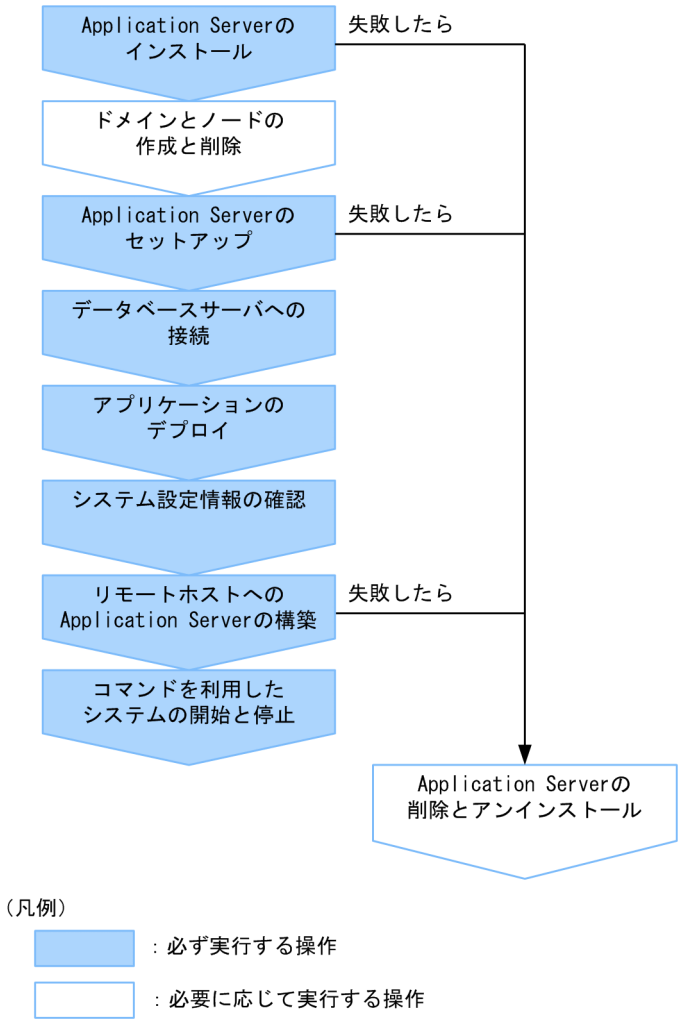
  Administration Console にログインして、GUI を利用して構築します。ドメインに関する操作を除き、コマンドと同様の操作ができます。

関連項目

- 7.4.1　Administration Console にログインする

4.2　アプリケーション実行環境の構築の流れ

アプリケーション実行環境を構築するために実施する作業と、作業の流れについて説明します。アプリケーション実行環境を構築するには、Application Server をインストールし、アプリケーション実行環境をセットアップします。そのあと、データベースサーバと接続し、アプリケーションをデプロイします。さらに、リモートホストに Application Server を構築し、運用管理サーバを利用するための設定をします。すべての設定が完了したら、システムを停止します。コマンドでアプリケーション実行環境を構築する場合は、管理者権限で実行します。



関連項目

- 4.4.1　Application Server のインストールの種類について
- 4.5.1　ドメインを作成する
- 4.5.2　ノードを作成する
- 4.5.3　ノードを削除する
- 4.5.4　ドメインを削除する
- 4.6.1　Application Server のセットアップの流れ

- 4.7.1　データベースサーバへの接続の流れ
- 4.8.1　アプリケーションのデプロイの流れ
- 4.9.1　システム設定情報の確認について
- 4.10.1　リモートホストへのアプリケーション実行環境の構築の流れ
- 7.2.1　コマンドを利用してシステムを開始する
- 7.2.2　コマンドを利用してシステムを停止する
- 4.11.1　Application Server を削除する
- 4.11.2　Application Server をアンインストールする

4.3　ディスク占有量およびメモリー所要量について

Application Server のディスク占有量、Application Server で使用するメモリー所要量について説明します。

ディスク占有量

AIX の場合

AIX の場合のディスク占有量については、リリースノートを参照してください。

Linux の場合

機能名	ディスク占有量（MB）
Application Server - Base	44.0
Developer's Kit for Java	179.0
Java EE Server	140.0
Web Server	36.0

メモリー所要量

AIX の場合

AIX の場合のメモリー所要量については、リリースノートを参照してください。

Linux の場合

機能名	メモリー所要量（MB）
Application Server - Base	20
Developer's Kit for Java	各 Java アプリケーションに含まれます
Java EE Server（サーバインスタンス分）	2,649
Java EE Server（ドメイン管理サーバ分）	1,000
Web Server	90

4.4 Application Server のインストール

Application Server のインストール手順について説明します。Application Server は、OS の準備が完了しているマシンに、製品の提供媒体からインストーラーを利用してインストールします。

4.4.1 Application Server のインストールの種類について

インストールの種類には、Application Server がインストールされていないマシンにインストールする新規インストール、複数の Application Server を別のディレクトリーにインストールする複数インストール、V9 以前のアプリケーション実行環境、または V9 以前のアプリケーション開発環境がインストールされているディレクトリーとは別のディレクトリーに Application Server をインストールする追加インストール、および Application Server をアップデートする上書きインストールがあります。

Application Server のインストールには、次の種類があります。

新規インストール

Application Server がインストールされていないマシンで、Application Server をインストールすることです。

複数インストール

Application Server がインストール済みのマシンで、Application Server を別のディレクトリーにインストールすることです。

マシン内に Application Server を共存させる運用をする場合に使用します。例えば、1 台のマシンに異なるバージョンの Application Server を共存させて、アップデート前の実行環境を残したまま、アップデート後の実行環境へ移行することもできます。

追加インストール

V9 以前のアプリケーション実行環境、または V9 以前のアプリケーション開発環境がインストール済みのマシンで、V10 以降の Application Server を別のディレクトリーにインストールすることです。

マシン内に異なるバージョン、または異なるエディションの Application Server を共存させる運用をする場合に使用します。例えば、V9 以前のアプリケーション開発環境が構築済みのマシンで、V10 以降の Application Server を別ディレクトリーに追加インストールできます。

上書きインストール

Application Server がインストール済みのマシンで、アップデートする Application Server を同じディレクトリーにインストールすることです。インストール済みの Application Server は上書きされます。

関連項目

- 4.1 構築するアプリケーション実行環境について
- 4.4.2 Application Server を新規インストールする
- 4.4.3 Application Server を複数インストールする

4.4.2　Application Server を新規インストールする

Application Server を新規インストールするには、製品の提供媒体から日立 PP インストーラーを起動し、PP インストール画面でインストールするプログラムを選択します。インストール完了後、環境変数の設定をします。

前提条件

- 管理者特権（スーパーユーザー）で実行する
- 前提としている OS（OS のパッチを含む）のインストールが完了している
- Application Server がインストールされていない

想定ユーザー

- システム構築者

操作手順

1. 次の表に示すパッケージがインストールされていることを確認します。

表 4-1　OS ごとに適用するパッケージ

パッケージ名	Red Hat Enterprise Linux 5	Red Hat Enterprise Linux Server 6	Red Hat Enterprise Linux Server 7
compat-libstdc++-296 (i386)	○	—	—
compat-libstdc++-33 (i386)	○	—	—
compat-libstdc++-33 (i686)	—	○	—
compat-libstdc++-33 (x86_64)	○	○	—
coreutils (x86_64)	○	○	○
findutils (x86_64)	○	○	○
gdb (x86_64)	○	○	○
glibc (i686)	○	○	○※1
glibc (i686) 2.5-24 以降	○※2	—	—
glibc (x86_64)	—	○	○
glibc (x86_64) 2.5-24 以降	○※2	—	—

パッケージ名	Red Hat Enterprise Linux 5	Red Hat Enterprise Linux Server 6	Red Hat Enterprise Linux Server 7
glibc-common (x86_64)	—	○	○
glibc-common (x86_64) 2.5-24 以降	○※2	—	—
glibc-devel (i386) 2.5-24 以降	○※2	—	—
glibc-devel (i686)	—	○	○※1
glibc-devel (x86_64)	—	○	○
glibc-devel (x86_64) 2.5-24 以降	○※2	—	—
glibc-headers (x86_64)	—	○	○
glibc-headers (x86_64) 2.5-24 以降	○※2	—	—
glibc-utils (x86_64)	—	○	○※1
glibc-utils (x86_64) 2.5-24 以降	○※2	—	—
gzip (x86_64)	○	○	○
ksh (x86_64)	—	○	○※1
libgcc (i386)	○	—	—
libgcc (i686)	—	○	○※1
libstdc++ (i386)	○	—	—
libstdc++ (i686)	—	○	○※1
lksctp-tools (x86_64)	○	○	○※1
ncompress (x86_64)	○	○※3	○※1
ncurses (x86_64)	○	○	○
net-tools (x86_64)	○	○	○
nscd (x86_64)	—	○	○※1
nscd (x86_64) 2.5-24 以降	○※2	—	—
nss-softokn-freebl (i686)	—	○	○※1
procps (x86_64)	○	○	○
rpm (x86_64)	○	○	○
sysstat (x86_64)	○	○	○
tar (x86_64)	○	○	○
tcsh (x86_64)	○	○	○※1

(凡例)

○：適用します。

—：該当しません。

注

AIX のパッケージについては、リリースノートを参照してください。

注※1

これらのパッケージについては、使用している Red Hat Enterprise Linux Server 7 のバージョンによってはデフォルトでインストールされない場合があります。

注※2

これらのパッケージについては、使用している Red Hat Enterprise Linux 5 のバージョンによってはインストールメディアに含まれていない場合があります。

注※3

Red Hat Enterprise Linux Server 6 のバージョンが 6.2 の場合、ncompress (x86_64)のバージョンは 4.2.4-54.el6_2.1 以降です。

パッケージがインストールされているかどうかは、rpm コマンドを実行して確認します。コマンドの実行例と実行結果の例を次に示します。

(実行例)

```
#rpm -q --qf  '%{NAME}-%{ARCH}¥n'  ncompress
```

(実行結果の例)

```
ncompress-x86_64
```

この例では、64 ビット版の ncompress パッケージがインストールされていることを示します。

「package *パッケージ名* is not installed」メッセージが表示された場合は、パッケージがインストールされていないので、*パッケージ名*に示すパッケージをインストールしてください。なお、インストール時には、関連するパッケージも必要に応じてインストールしてください。

2. 日立 PP インストーラー実行時の言語種別と、実行するターミナルの言語が一致しているかどうかを確認し、一致していない場合は一致させます。

3. 製品の提供媒体を該当するドライブにセットします。

4. mount コマンドを実行して、該当する媒体のファイルシステムをマウントします。

AIX の場合

```
mount -r -v cdrfs デバイススペシャルファイル名
該当する媒体のファイルシステムのマウントディレクトリー名
```

Linux の場合

```
mount -r -o mode=0544 /dev/cdrom
該当する媒体のファイルシステムのマウントディレクトリー名
```

デバイススペシャルファイル名、および*該当する媒体のファイルシステムのマウントディレクトリー名*は、OS、ハードウェア、および環境によって異なります。

5. 該当する媒体のセットアッププログラムを起動します。

AIX の場合

```
該当する媒体のファイルシステムのマウントディレクトリー名/AIX/setup -m
該当する媒体のファイルシステムのマウントディレクトリー名
```

Linux の場合

```
該当する媒体のファイルシステムのマウントディレクトリー名/x64lin/setup -m
該当する媒体のファイルシステムのマウントディレクトリー名
```

重要

該当する媒体のディレクトリー名やファイル名は、マシン環境によって記述した内容と見え方が異なることがあります。ls コマンドで確認し、表示されたファイル名をそのまま入力してください。

該当する媒体のセットアッププログラムによって、日立 PP インストーラーと常駐プロセス自動起動プログラムがハードディスク上にインストールされ、日立 PP インストーラーが自動的に起動されます。

6. 日立 PP インストーラーのメインメニューで、[I] キーを押します。

7. PP インストール画面で、インストールする製品（Hitachi Application Server）にカーソルを移動させ、[スペース] キーを押します。

製品名の左側には、選択状態を示す<@>が表示されます。

8. 製品を構成するプログラムを選択してインストールする場合、インストールしないプログラムにカーソルを移動させ、[スペース] キーを押します。

インストールが任意のプログラムの左側には、[@]が表示されます。[スペース] キーを押すと、[ ]に表示が変わり、インストール対象から外されます。

なお、インストールが必須のプログラムの左側には、@が表示されます。

9. インストールする製品の左側に<@>、およびプログラムの左側に[@]または@が表示されていることを確認して、[I] キーを押します。

画面の最下行に、Install PP? (y: install, n: cancel)==>が表示されます。

10. [y] キーまたは [Y] キーを押します。

画面の最下行に、Enter the installation path ==>*インストール先のデフォルト値*が表示されます。

[Ctrl] ＋ [n] キーまたは [Ctrl] ＋ [N] キーを押すと、インストールが中止されて PP インストール画面に戻ります。

11. *インストール先のデフォルト値*を確認し、必要に応じてインストール先を変更して、[Enter] キーを押します。

インストール先には半角英数字、-、_以外は指定できません。

インストール先ディレクトリーが存在しない場合、ディレクトリー作成を確認するメッセージが表示されます。

```
The specified path does not exist. Do you want to create the path?
Specified path: インストール先
```

*インストール先*を確認して、[y] キーまたは [Y] キーを押してください。

画面の最下行に、Enter a display name ==>*表示名のデフォルト値*が表示されます。

インストール先を再設定する場合は、[Ctrl] + [B] キーを押します。

12. *表示名のデフォルト値*を確認し、必要に応じて表示名を変更して、[Enter] キーを押します。

表示名に指定できる文字の長さは、半角で 40 文字です。ただし、半角英数字、-、_以外は指定できません。

画面に、指定した表示名、およびインストール先でインストールを続行してよいかどうかを確認するメッセージが表示されます。

```
This program product will be installed at the specified path,
with the specified display name.
Do you want to continue?
Specified display name: 表示名
Specified path        : インストール先
```

表示名、およびインストール先を再設定する場合は、[n] キーまたは [N] キーを押すと、手順 10 に戻ります。

13. 表示名、およびインストール先を確認して、[y] キーまたは [Y] キーを押します。

インストールが開始されます。

14. インストール終了を示すメッセージが出力されたら、[Q] キーを押します。

15. 日立 PP インストーラーのメインメニューで、[L] キーを押します。

PP 一覧表示画面にインストール済みの製品一覧が表示されます。

16. 製品がインストールされていることを確認して、[Q] キーを押します。

17. 日立 PP インストーラーのメインメニューで、[Q] キーを押します。

日立 PP インストーラーが終了し、Application Server のインストールが完了します。

次の作業

- インストール後の環境設定をする

関連項目

- 4.4.6　インストール後の環境設定をする
- 4.1　構築するアプリケーション実行環境について

4.4.3　Application Server を複数インストールする

Application Server を同じマシンの別のディレクトリーに複数インストールするには、製品の提供媒体から、-m オプションを指定して日立 PP インストーラーを起動し、PP インストール画面でインストールするプログラムを選択します。

前提条件

- 管理者特権（スーパーユーザー）で実行する
- 前提としている OS（OS のパッチを含む）のインストールが完了している
- 同じエディションの Application Server が 1 つ以上インストールされている

想定ユーザー

- システム構築者

操作手順

1. 製品の提供媒体を該当するドライブにセットします。
2. mount コマンドを実行して、該当する媒体のファイルシステムをマウントします。

AIX の場合

```
mount -r -v cdrfs デバイススペシャルファイル名
該当する媒体のファイルシステムのマウントディレクトリー名
```

Linux の場合

```
mount -r -o mode=0544 /dev/cdrom
該当する媒体のファイルシステムのマウントディレクトリー名
```

デバイススペシャルファイル名、および*該当する媒体のファイルシステムのマウントディレクトリー名*は、OS、ハードウェア、および環境によって異なります。

3. 該当する媒体のセットアッププログラムを起動します。

AIX の場合

```
該当する媒体のファイルシステムのマウントディレクトリー名/AIX/setup -m
該当する媒体のファイルシステムのマウントディレクトリー名
```

Linux の場合

```
該当する媒体のファイルシステムのマウントディレクトリー名/x64lin/setup -m
該当する媒体のファイルシステムのマウントディレクトリー名
```

重要

該当する媒体のディレクトリー名やファイル名は、マシン環境によって記述した内容と見え方が異なることがあります。ls コマンドで確認し、表示されたファイル名をそのまま入力してください。

該当する媒体のセットアッププログラムによって、日立 PP インストーラーと常駐プロセス自動起動プログラムがハードディスク上にインストールされ、日立 PP インストーラーが自動的に起動されます。

4. 日立 PP インストーラーのメインメニューで、[L] キーを押します。
   PP 一覧表示画面にインストール済みの製品一覧が表示されます。

5. 製品がインストールされていることを確認して、[Q] キーを押します。

6. 日立 PP インストーラーのメインメニューで、[I] キーを押します。

7. PP インストール画面で、インストールする製品（Hitachi Application Server）にカーソルを移動させ、[スペース] キーを押します。
   製品名の左側には、選択状態を示す<@>が表示されます。

8. 製品を構成するプログラムを選択してインストールする場合、インストールしないプログラムにカーソルを移動させ、[スペース] キーを押します。
   インストールが任意のプログラムの左側には、[@]が表示されます。[スペース] キーを押すと、[ ]に表示が変わり、インストール対象から外されます。
   なお、インストールが必須のプログラムの左側には、@が表示されます。

9. インストールする製品の左側に<@>、およびプログラムの左側に[@]または@が表示されていることを確認して、[I] キーを押します。
   画面の最下行に、Install PP? (y: install, n: cancel)==>が表示されます。

10. [y] キーまたは [Y] キーを押します。
    画面の最下行に、Enter the installation path ==>*インストール先のデフォルト値*が表示されます。
    [Ctrl] ＋ [n] キーまたは [Ctrl] ＋ [N] キーを押すと、インストールが中止されて PP インストール画面に戻ります。

11. *インストール先のデフォルト値*を確認し、必要に応じてインストール先を変更して、[Enter] キーを押します。
    *インストール先のデフォルト値*には、手順 5 で確認したインストール先と異なる値を指定してください。
    インストール先には半角英数字、-、_以外は指定できません。

インストール先ディレクトリーが存在しない場合、ディレクトリー作成を確認するメッセージが表示されます。

```
The specified path does not exist. Do you want to create the path?
Specified path: インストール先
```

*インストール先*を確認して、[y] キーまたは [Y] キーを押してください。

画面の最下行に、Enter a display name ==>*表示名のデフォルト値*が表示されます。

インストール先を再設定する場合は、[Ctrl] ＋ [B] キーを押します。

12. *表示名のデフォルト値*を確認し、必要に応じて表示名を変更して、[Enter] キーを押します。

*表示名のデフォルト値*には、手順 5 で確認した表示名と異なる値を指定してください。

表示名に指定できる文字の長さは、半角で 40 文字です。

画面に、指定した表示名、およびインストール先でインストールを続行してよいかどうかを確認するメッセージが表示されます。

```
This program product will be installed at the specified path,
with the specified display name.
Do you want to continue?
Specified display name: 表示名
Specified path        : インストール先
```

表示名、およびインストール先を再設定する場合は、[n] キーまたは [N] キーを押すと、手順 10 に戻ります。

13. 表示名、およびインストール先を確認して、[y] キーまたは [Y] キーを押します。

インストールが開始されます。

14. インストール終了を示すメッセージが出力されたら、[Q] キーを押します。

15. 日立 PP インストーラーのメインメニューで、[L] キーを押します。

PP 一覧表示画面にインストール済みの製品一覧が表示されます。

16. 製品がインストールされていることを確認して、[Q] キーを押します。

17. 日立 PP インストーラーのメインメニューで、[Q] キーを押します。

日立 PP インストーラーが終了し、Application Server のインストールが完了します。

次の作業

- インストール後の環境設定をする

関連項目

- 4.4.6　インストール後の環境設定をする
- 4.1　構築するアプリケーション実行環境について

4.4.4 V9 以前のアプリケーション実行環境が構築済みのマシンに Application Server を追加インストールする

V9 以前のアプリケーション実行環境が構築済みのマシンに、Application Server を同じマシンの別のディレクトリーに追加インストールするには、製品の提供媒体から、-m オプションを指定して日立 PP インストーラーを起動し、PP インストール画面でインストールするプログラムを選択します。

前提条件

- 管理者特権（スーパーユーザー）で実行する
- 前提としている OS（OS のパッチを含む）のインストールが完了している
- V9 以前のアプリケーション実行環境が構築されている

想定ユーザー

- システム構築者

操作手順

1. 製品の提供媒体を該当するドライブにセットします。

2. mount コマンドを実行して、該当する媒体のファイルシステムをマウントします。

   AIX の場合

   ```
   mount -r -v cdrfs デバイススペシャルファイル名
   該当する媒体のファイルシステムのマウントディレクトリー名
   ```

   Linux の場合

   ```
   mount -r -o mode=0544 /dev/cdrom
   該当する媒体のファイルシステムのマウントディレクトリー名
   ```

   デバイススペシャルファイル名、および*該当する媒体のファイルシステムのマウントディレクトリー名*は、OS、ハードウェア、および環境によって異なります。

3. 該当する媒体のセットアッププログラムを起動します。

   AIX の場合

   ```
   該当する媒体のファイルシステムのマウントディレクトリー名/AIX/setup -m
   該当する媒体のファイルシステムのマウントディレクトリー名
   ```

   Linux の場合

   ```
   該当する媒体のファイルシステムのマウントディレクトリー名/x64lin/setup -m
   該当する媒体のファイルシステムのマウントディレクトリー名
   ```

重要

該当する媒体のディレクトリー名やファイル名は、マシン環境によって記述した内容と見え方が異なることがあります。ls コマンドで確認し、表示されたファイル名をそのまま入力してください。

該当する媒体のセットアッププログラムによって、日立 PP インストーラーと常駐プロセス自動起動プログラムがハードディスク上にインストールされ、日立 PP インストーラーが自動的に起動されます。

4. 日立 PP インストーラーのメインメニューで、[I] キーを押します。

5. PP インストール画面で、インストールする製品（Hitachi Application Server）にカーソルを移動させ、[スペース] キーを押します。

製品名の左側には、選択状態を示す<@>が表示されます。

6. 製品を構成するプログラムを選択してインストールする場合、インストールしないプログラムにカーソルを移動させ、[スペース] キーを押します。

インストールが任意のプログラムの左側には、[@]が表示されます。[スペース] キーを押すと、[ ]に表示が変わり、インストール対象から外されます。

なお、インストールが必須のプログラムの左側には、@が表示されます。

7. インストールする製品の左側に<@>、およびプログラムの左側に[@]または@が表示されていることを確認して、[I] キーを押します。

画面の最下行に、Install PP? (y: install, n: cancel)==>が表示されます。

8. [y] キーまたは [Y] キーを押します。

画面の最下行に、Enter the installation path ==>*インストール先のデフォルト値*が表示されます。

[Ctrl] ＋ [n] キーまたは [Ctrl] ＋ [N] キーを押すと、インストールが中止されて PP インストール画面に戻ります。

9. *インストール先のデフォルト値*を確認し、必要に応じてインストール先を変更して、[Enter] キーを押します。

*インストール先のデフォルト値*には、V9 以前のアプリケーション実行環境のインストール先の/opt/Cosminexus と異なる値を指定してください。

インストール先には半角英数字、-、_以外は指定できません。

インストール先ディレクトリーが存在しない場合、ディレクトリー作成を確認するメッセージが表示されます。

```
The specified path does not exist. Do you want to create the path?
Specified path: インストール先
```

*インストール先*を確認して、[y] キーまたは [Y] キーを押してください。

画面の最下行に、Enter a display name ==>*表示名のデフォルト値*が表示されます。

インストール先を再設定する場合は、[Ctrl] ＋ [B] キーを押します。

10. *表示名のデフォルト値*を確認し、必要に応じて表示名を変更して、[Enter] キーを押します。

表示名に指定できる文字の長さは、半角で 40 文字です。ただし、半角英数字、-、_以外は指定できません。

画面に、指定した表示名、およびインストール先でインストールを続行してよいかどうかを確認するメッセージが表示されます。

```
This program product will be installed at the specified path,
with the specified display name.
Do you want to continue?
Specified display name: 表示名
Specified path        : インストール先
```

表示名、およびインストール先を再設定する場合は、[n] キーまたは [N] キーを押すと、手順 8 に戻ります。

11. 表示名、およびインストール先を確認して、[y] キーまたは [Y] キーを押します。

インストールが開始されます。

12. インストール終了を示すメッセージが出力されたら、[Q] キーを押します。

13. 日立 PP インストーラーのメインメニューで、[L] キーを押します。

PP 一覧表示画面にインストール済みの製品一覧が表示されます。

14. 製品がインストールされていることを確認して、[Q] キーを押します。

15. 日立 PP インストーラーのメインメニューで、[Q] キーを押します。

日立 PP インストーラーが終了し、Application Server のインストールが完了します。

次の作業

- インストール後の環境設定をする

関連項目

- 4.4.6　インストール後の環境設定をする
- 4.1　構築するアプリケーション実行環境について

4.4.5　Application Server を上書きインストールする

Application Server を同じマシンの同じディレクトリーに上書きインストールするには、製品の提供媒体から日立 PP インストーラーを起動し、PP インストール画面でインストールするプログラムを選択します。

前提条件

- 管理者特権スーパーユーザー）で実行する

- 前提としている OS（OS のパッチを含む）のインストールが完了している
- アップデートする Application Server と同じ Application Server がインストールされている
- アップデートする Application Server を使用したシステムが停止している

想定ユーザー

- システム構築者

操作手順

1. インストール済みの Application Server 上に、稼働中のサーバやドメインがある場合は、次の手順で停止します。
   a. asadmin ユーティリティーコマンドのstop-servers サブコマンドを実行して、Application Server を停止します。

   ```
   asadmin stop-servers
   ```

   コマンドの実行結果を次に示します。

   ```
   Command stop-servers executed successfully.
   ```

   b. asadmin ユーティリティーコマンドの stop-domain サブコマンドを実行して、ドメインを停止します。

   ```
   asadmin stop-domain ドメイン名
   ```

   コマンドの実行結果を次に示します。

   ```
   Command stop-domain executed successfully.
   ```

   c. asadmin ユーティリティーコマンドの list-domains サブコマンドを実行して、ドメインの一覧を表示します。

   ```
   asadmin list-domains
   ```

   コマンドの実行結果を次に示します。停止したドメインのステータスが not running になっていることを確認してください。

   ```
   domain1 not running
   ドメイン名 not running
   Command list-domains executed successfully.
   ```

2. 製品の提供媒体を該当するドライブにセットします。

3. mount コマンドを実行して、該当する媒体のファイルシステムをマウントします。

AIX の場合

```
mount -r -v cdrfs デバイススペシャルファイル名
該当する媒体のファイルシステムのマウントディレクトリー名
```

Linux の場合

```
mount -r -o mode=0544 /dev/cdrom
該当する媒体のファイルシステムのマウントディレクトリー名
```

デバイススペシャルファイル名、および*該当する媒体のファイルシステムのマウントディレクトリー名*は、OS、ハードウェア、および環境によって異なります。

4. 該当する媒体のセットアッププログラムを起動します。

AIX の場合

```
該当する媒体のファイルシステムのマウントディレクトリー名/AIX/setup -m
該当する媒体のファイルシステムのマウントディレクトリー名
```

Linux の場合

```
該当する媒体のファイルシステムのマウントディレクトリー名/x64lin/setup -m
該当する媒体のファイルシステムのマウントディレクトリー名
```

重要

該当する媒体のディレクトリー名やファイル名は、マシン環境によって記述した内容と見え方が異なることがあります。ls コマンドで確認し、表示されたファイル名をそのまま入力してください。

該当する媒体のセットアッププログラムによって、日立 PP インストーラーと常駐プロセス自動起動プログラムがハードディスク上にインストールされ、日立 PP インストーラーが自動的に起動されます。

5. 日立 PP インストーラーのメインメニューで、[L] キーを押します。

PP 一覧表示画面にインストール済みの製品一覧が表示されます。

6. 上書きインストールする製品のインストール先を確認して、[Q] キーを押します。

7. 日立 PP インストーラーのメインメニューで、[I] キーを押します。

8. PP インストール画面で、上書きインストールする製品にカーソルを移動させ、[スペース] キーを押します。

製品名の左側には、選択状態を示す<@>が表示されます。

9. 製品を構成するプログラムを選択して上書きインストールする場合、インストールしないプログラムにカーソルを移動させ、[スペース] キーを押します。

   インストールが任意のプログラムの左側には、[@]が表示されます。[スペース] キーを押すと、[ ]に表示が変わり、上書きインストール対象から外されます。

   なお、インストールが必須のプログラムの左側には、@が表示されます。

10. 上書きインストールする製品の左側に<@>、およびプログラムの左側に[@]または@が表示されていることを確認して、[I] キーを押します。

    画面の最下行に、Install PP? (y: install, n: cancel)==>が表示されます。

11. [y] キーまたは [Y] キーを押します。

    画面の最下行に、Enter the installation path ==>*インストール先*が表示されます。

    [Ctrl] ＋ [n] キーまたは [Ctrl] ＋ [N] キーを押すと、インストールが中止されて PP インストール画面に戻ります。

12. *インストール先*を手順 6 で確認したインストール先に変更して、[Enter] キーを押します。

    指定した*インストール先*に同一製品がインストール済みであるが、インストールを実行してよいかどうかを確認するメッセージが表示されます。

```
This program product is already installed.
Do you want to continue with the installation?
Installation path: インストール先
```

    インストール先を再設定する場合は、[n] キーまたは [N] キーを押します。

13. [y] キーまたは [Y] キーを押します。

    画面に、指定した表示名、およびインストール先でインストールを続行してよいかどうかを確認するメッセージが表示されます。

```
This program product will be installed at the specified path,
with the specified display name.
Do you want to continue?
Specified display name: 表示名
Specified path        : インストール先
```

    インストール先を再設定する場合は、[n] キーまたは [N] キーを押すと、手順 11 に戻ります。

14. *設定済みの表示名*を確認し、[y] キーまたは [Y] キーを押します。

    インストールが開始されます。

15. インストール終了を示すメッセージが出力されたら、[Q] キーを押します。

16. 日立 PP インストーラーのメインメニューで、[L] キーを押します。

    PP 一覧表示画面にインストール済みの製品一覧が表示されます。

17. 製品がインストールされていることを確認して、[Q] キーを押します。

18. 日立 PP インストーラーのメインメニューで、［Q］キーを押します。

日立 PP インストーラーが終了し、Application Server の上書きインストールが完了します。

関連項目

- 4.1　構築するアプリケーション実行環境について

4.4.6　インストール後の環境設定をする

Application Server のインストール後に、システムが動作するように環境設定を行います。

前提条件

- Application Server がインストールされている

想定ユーザー

- システム構築者

操作手順

1. 設定ファイル（asenv.conf）に TZ 環境変数を設定します。

設定ファイル（asenv.conf）は、*Java EE Server のインストールディレ クトリー*/glassfish/config/asenv.conf に格納されています。設定ファイル（asenv.conf）の定義例を次に示します。

```
TZ=JST-9
```

2. /etc/hosts ファイルに、自ホスト名に対してネットワークインターフェースに割り当てられた適切な IP アドレスを設定して、自ホスト名から IP アドレスを解決できるように設定します。

4.5　ドメインとノードの作成と削除

コマンドを利用してドメインとノードを作成する手順について説明します。また、コマンドを利用して不要なドメインとノードを削除する手順についても説明します。なお、デフォルトのドメインとノードは、Application Server のインストールが完了すると作成されます。

4.5.1　ドメインを作成する

ドメインを作成するには、asadmin ユーティリティーコマンドのcreate-domain サブコマンドを実行します。なお、デフォルトのドメインはdomain1 です。

前提条件

- Application Server のインストールが完了している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのcreate-domain サブコマンドを実行して、ドメインを作成します。

```
asadmin create-domain --adminport Adminのポート --instanceport HTTPのポート
--domainproperties domain.jmxPort=値:http.ssl.port=値:java.debugger.port=値:
jms.port=値:orb.listener.port=値:orb.mutualauth.port=値:orb.ssl.port=値:
osgi.shell.telnet.port=値 ドメイン名
```

> **重要**
>
> *Admin のポート*は、作成したドメインに対してasadmin ユーティリティーコマンドのサブコマンドを実行する際に、asadmin ユーティリティーオプションの--port オプションで指定する必要があります。

コマンド実行時に、管理ユーザーのユーザー名の入力要求（Enter admin user name [Enter to accept default ""admin"" / no password]）に対してユーザー名、管理ユーザーのパスワードの入力要求（Enter the admin password [Enter to accept default of no password]）に対してパスワードを設定します。なお、管理ユーザーのユーザー名のデフォルト値はadmin、パスワードのデフォルト値はありません。

コマンドの実行結果を次に示します。

```
Command create-domain executed successfully.
```

2. asadmin ユーティリティーコマンドのstart-domain サブコマンドを実行して、ドメインを起動します。

```
asadmin start-domain ドメイン名
```

コマンドの実行結果を次に示します。

```
Command start-domain executed successfully.
```

3. asadmin ユーティリティーコマンドのlist-domains サブコマンドを実行して、ドメインの一覧を表示します。

```
asadmin list-domains
```

コマンドの実行結果を次に示します。手順 2 で起動したドメインのステータスがrunning になっていることを確認してください。

```
domain1 running
ドメイン名 running
Command list-domains executed successfully.
```

次の作業

- ノードを作成する

関連項目

- 4.5.2　ノードを作成する

4.5.2　ノードを作成する

ノードを作成するには、asadmin ユーティリティーコマンドのcreate-node-config サブコマンドを実行します。なお、デフォルトのノードはlocalhost-domain1 です。

前提条件

- Application Server のインストールが完了している
- 作成したドメイン管理サーバが起動している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのcreate-node-config サブコマンドを実行して、ノードを作成します。

```
asadmin create-node-config
--nodehost ノードに対するホスト名
--installdir Application Serverのインストールディレクトリー/javaeeの絶対パス
--nodedir ノードの情報を格納するディレクトリーのパス ノード名
```

> **重要**
>
> デフォルトのドメイン以外にノードを作成する場合は、asadmin ユーティリティーオプションの--port オプションで*Admin のポート*を指定して、create-node-config サブコマンドを実行してください。

コマンドの実行結果を次に示します。

```
Command create-node-config executed successfully.
```

4.5.3 ノードを削除する

ノードを削除するには、asadmin ユーティリティーコマンドのdelete-node-config サブコマンドを実行します。

前提条件

- Application Server のインストールが完了している
- 作成したドメイン管理サーバが起動している
- ノードに対するホストで Application Server の削除が完了している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのlist-nodes サブコマンドを実行して、ノードの一覧を表示します。

```
asadmin list-nodes
```

重要

デフォルトのドメイン以外でノードの一覧を表示する場合は、asadmin ユーティリティーオプションの--port オプションで*Admin のポート*を指定して、list-nodes サブコマンドを実行してください。

コマンドの実行結果を次に示します。

```
ノード名　CONFIG　IPアドレスまたはホスト名
ノード名　CONFIG　IPアドレスまたはホスト名
Command list-nodes executed successfully.
```

2. asadmin ユーティリティーコマンドのdelete-node-config サブコマンドを実行して、ノードを削除します。

```
asadmin delete-node-config ノード名
```

重要

- デフォルトのドメイン以外でノードを削除する場合は、asadmin ユーティリティーオプションの--port オプションで*Admin のポート*を指定して、delete-node-config サブコマンドを実行してください。
- 削除するノードのノードディレクトリー以下のファイルまたはディレクトリーをほかのプロセスが使用していると、ノードの削除に失敗する場合があります。稼働中のほかのプログラムでノードディレクトリー以下のファイルまたはディレクトリーを使用していないことを確認してから、ノードの削除を実行してください。

コマンドの実行結果を次に示します。

```
Command delete-node-config executed successfully.
```

4.5.4　ドメインを削除する

ドメインを削除するには、asadmin ユーティリティーコマンドのstop-domain サブコマンドを実行してドメインを停止し、delete-domain サブコマンドを実行します。

前提条件

- Application Server のインストールが完了している
- 作成したドメイン管理サーバが起動している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのstop-domain サブコマンドを実行して、ドメインを停止します。

```
asadmin stop-domain ドメイン名
```

コマンドの実行結果を次に示します。

```
Command stop-domain executed successfully.
```

2. asadmin ユーティリティーコマンドのlist-domains サブコマンドを実行して、ドメインの一覧を表示します。

```
asadmin list-domains
```

コマンドの実行結果を次に示します。手順 1 で停止したドメインのステータスがnot running になっていることを確認してください。

```
domain1 running
ドメイン名 not running
Command list-domains executed successfully.
```

3. asadmin ユーティリティーコマンドのdelete-domain サブコマンドを実行して、ドメインを削除します。

```
asadmin delete-domain ドメイン名
```

> **重要**
>
> 削除するドメインのドメインディレクトリー以下のファイルまたはディレクトリーをほかのプロセスが使用していると、ドメインの削除に失敗する場合があります。稼働中のほかのプログラムでドメインディレクトリー以下のファイルまたはディレクトリーを使用していないことを確認してから、ドメインの削除を実行してください。

コマンドの実行結果を次に示します。

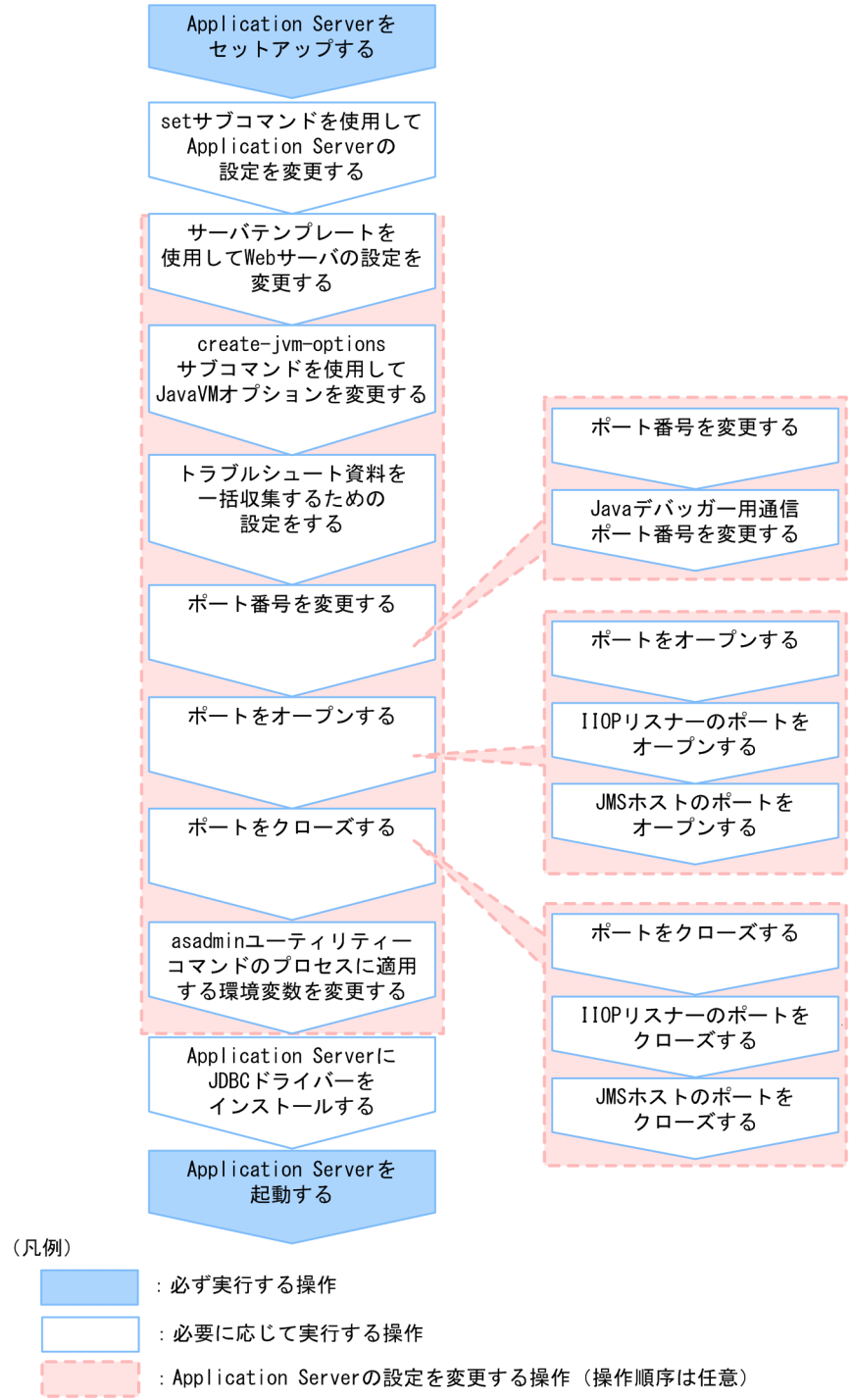
```
Command delete-domain executed successfully.
```

4.6　Application Server のセットアップ

Application Server をセットアップする手順について説明します。Application Server をセットアップすると、ホスト上にパフォーマンストレーサー、Java EE サーバ、および Web サーバが構築されます。また、コマンドを利用して Application Server を起動する手順についても説明します。

4.6.1　Application Server のセットアップの流れ

Application Server をセットアップするために実施する作業と、作業の流れについて説明します。
Application Server のセットアップでは、ホスト上にパフォーマンストレーサー、Java EE サーバ、および Web サーバを構築し、各サーバのサーバ間関連を設定します。そのあと、Application Server の設定を変更して、Application Server を起動します。



関連項目

- 4.6.2　Application Server をセットアップする
- 4.6.3　set サブコマンドを使用して Application Server の設定を変更する
- 4.6.4　サーバテンプレートを使用して Web サーバの設定を変更する
- 4.6.5　create-jvm-options サブコマンドを使用して JavaVM オプションを変更する
- 4.6.6　トラブルシュート資料を一括収集するための設定をする
- 4.6.7　ポート番号を変更する

4.6.2　Application Server をセットアップする

Application Server をセットアップするには、asadmin ユーティリティーコマンドのcreate-prf サブコマンドでパフォーマンストレーサー、create-cluster サブコマンドとcreate-instance サブコマンドで Java EE サーバ、およびcreate-webserver サブコマンドで Web サーバを構築します。そのあと、create-relation サブコマンドで各サーバのサーバ間関連を設定します。

前提条件

- Application Server のインストールが完了している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのstart-domain サブコマンドを実行して、ドメイン管理サーバを起動します。

```
asadmin start-domain
```

コマンドの実行結果を次に示します。

```
Command start-domain executed successfully.
```

2. asadmin ユーティリティーコマンドのcreate-prf サブコマンドを実行して、パフォーマンストレーサーを構築します。

```
asadmin create-prf --node ノード名 パフォーマンストレーサー名
```

コマンドの実行結果を次に示します。

```
Command create-prf executed successfully.
```

3. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

コマンドの実行結果を次に示します。手順 2 で構築したパフォーマンストレーサー名が表示されていることを確認してください。

```
パフォーマンストレーサー名 not running
Command list-prfs executed successfully.
```

4. 複数の Java EE サーバを配置するクラスター構成の場合、asadmin ユーティリティーコマンドのcreate-cluster サブコマンドを実行して、Java EE サーバをグループ化するクラスターを構築します。

```
asadmin create-cluster クラスター名
```

コマンドの実行結果を次に示します。

```
Command create-cluster executed successfully.
```

5. asadmin ユーティリティーコマンドのcreate-instance サブコマンドを実行して、Java EE サーバ（サーバインスタンス）を構築します。

```
asadmin create-instance --node ノード名 --prf パフォーマンストレーサー名
--cluster クラスター名 サーバインスタンス名
```

- *パフォーマンストレーサー名*には、手順 2 で構築したパフォーマンストレーサー名を指定します。
- 複数の Java EE サーバを配置するクラスター構成の場合、--cluster オプションには、手順 4 で構築したクラスター名を指定します。
- --prf オプションを指定することで、サーバインスタンスとパフォーマンストレーサーの間に関連（関連タイプ：prf-relation）が作成されます。

コマンドの実行結果を次に示します。

```
Command create-instance executed successfully.
```

6. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、Java EE サーバの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。手順 5 で構築したサーバインスタンス名が表示されていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 not running
Command list-instances executed successfully.
```

- *クラスター名*は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。

7. asadmin ユーティリティーコマンドのcreate-webserver サブコマンドを実行して、Web サーバを構築します。

```
asadmin create-webserver --node ノード名
--prf パフォーマンストレーサー名 Webサーバ名
```

- *パフォーマンストレーサー名*には、手順 2 で構築したパフォーマンストレーサー名を指定します。
- --prf オプションを指定することで、Web サーバとパフォーマンストレーサーの間に関連（関連タイプ：prf-relation）が作成されます。

コマンドの実行結果を次に示します。

```
Command create-webserver executed successfully.
```

8. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。手順 7 で構築した Web サーバ名が表示されていることを確認してください。

```
Webサーバ名 not running
Command list-webservers executed successfully.
```

9. asadmin ユーティリティーコマンドのcreate-relation サブコマンドを実行して、Web サーバが受信したリクエストのリダイレクト先となる Java EE サーバ（サーバインスタンス）を関連づけます。

```
asadmin create-relation --relationtype redirect
--from Webサーバ名 --to サーバインスタンス名
[--properties サーバ間関連のプロパティー名=サーバ間関連のプロパティーの値
[:サーバ間関連のプロパティー名=サーバ間関連のプロパティーの値]...]
サーバ間関連の関連名
```

- *Web サーバ名*には、手順 7 で構築した Web サーバ名を指定します。
- *サーバインスタンス名*には、手順 5 で構築したサーバインスタンス名を指定します。
- リダイレクト関連の関連づけを設定する場合、静的コンテンツを Web サーバで処理し、静的コンテンツ以外のリクエストを Java EE サーバで処理するために、--properties オプションにpath、およびnetwork-listener を指定します。

  （例）

  path=/apserver/:network-listener=http-listener-1

  path には、スラッシュ（/）から始まる URL のパスを指定します。スラッシュだけの指定（path=/）はしないでください。この例では、http://xxxxxxxxxx/index.html のように URL のパスの先頭に apserver を含まないリクエストは、Web サーバの静的コンテンツにアクセスされます。また、http://xxxxxxxxxx/apserver/sample/index.jsp のように URL のパスの先頭にapserver を含むリ

クエストは、http://yyyyyyyyyy/sample/index.jsp という URL で Java EE サーバにリダイレクトされます。

network-listener には、リダイレクト先の Java EE サーバの HTTP リスナー、または HTTPS リスナーを示すネットワークリスナー名を指定します。Java EE サーバには、デフォルトで HTTP リスナーにhttp-listener-1、HTTPS リスナーにhttp-listener-2 というネットワークリスナーが定義されています。

コマンドの実行結果を次に示します。

```
Command create-relation executed successfully.
```

10. asadmin ユーティリティーコマンドのlist-relations サブコマンドを実行して、サーバ間関連の一覧を表示します。

```
asadmin list-relations
```

コマンドの実行結果を次に示します。次のサーバ間関連が表示されていることを確認してください。

- サーバインスタンスとパフォーマンストレーサーとのパフォーマンストレーサー関連
- Web サーバとパフォーマンストレーサーとのパフォーマンストレーサー関連
- 手順 9 で設定した Web サーバとサーバインスタンスとのリダイレクト関連

```
サーバ間関連の関連名  prf-relation サーバインスタンス名
パフォーマンストレーサー名
サーバ間関連の関連名  prf-relation Webサーバ名
パフォーマンストレーサー名
サーバ間関連の関連名  redirect Webサーバ名
サーバインスタンス名
Command list-relations executed successfully.
```

4.6.3 set サブコマンドを使用して Application Server の設定を変更する

Application Server の設定を変更するには、asadmin ユーティリティーコマンドのset サブコマンドを実行して、Application Server の設定値を変更します。

前提条件

- ドメイン管理サーバが起動している
- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのget サブコマンドを実行して、Application Server の設定値を取得します。

```
asadmin get "*"
```

コマンドの実行結果から、変更対象の設定対象識別子と、変更前の設定値を確認してください。

複数のパラメーターに値が設定されている場合、設定値は次のように扱われます。

パフォーマンストレーサー関連のパラメーターの場合

「hitachi-prfs.hitachi-prf.パフォーマンストレーサー名.で始まるパラメーター」と「hitachi-prf-configs.hitachi-prf-config.パフォーマンストレーサーのコンフィグ名で始まるパラメーター」の両方に値が設定されているときは、「hitachi-prfs.hitachi-prf.パフォーマンストレーサー名.で始まるパラメーター」の値が有効になります。

Web サーバ関連のパラメーターの場合

「hitachi-webservers.hitachi-webserver.Web サーバ名で始まるパラメーター」と「hitachi-webserver-configs.hitachi-webserver-config.Web サーバのコンフィグ名で始まるパラメーター」の両方に値が設定されているときは、「hitachi-webservers.hitachi-webserver.Web サーバ名で始まるパラメーター」の値が有効になります。

サーバインスタンス関連のパラメーターの場合

「servers.server.Java EE サーバ名で始まるパラメーター」と「configs.config.Java EE サーバの構成名で始まるパラメーター」の両方に値が設定されているときは、「servers.server.Java EE サーバ名で始まるパラメーター」の値が有効になります。

2. asadmin ユーティリティーコマンドのset サブコマンドを実行して、変更対象の設定対象識別子に対して変更後の設定値を指定します。

```
asadmin set 設定対象識別子=設定値
```

> **メモ**
>
> Web サーバの標準プロパティー以外（デイレクティブ）の設定値を変更する場合は、サーバテンプレートを使用して Web サーバの設定を変更してください。

コマンドの実行結果を次に示します。

```
Command set executed successfully.
```

3. asadmin ユーティリティーコマンドのget サブコマンドを実行して、Application Server の設定値を再度取得します。

```
asadmin get "*"
```

コマンドの実行結果から、変更対象の設定対象識別子に、手順 2 で指定した設定値が反映されていることを確認してください。

関連項目

- 4.6.4　サーバテンプレートを使用して Web サーバの設定を変更する

4.6.4　サーバテンプレートを使用して Web サーバの設定を変更する

Web サーバで標準プロパティー以外の設定値を変更するには、サーバテンプレートを使用します。サーバテンプレートには、Web サーバの運用に必要な設定を記述しています。Web サーバの設定を変更する場合は、拡張プロパティーを設定するか、またはディレクティブをサーバテンプレートに直接記述するかのどちらかを実施します。サーバテンプレートを編集する場合は、拡張プロパティーを設定する方法をお勧めします。

前提条件

- ドメイン管理サーバが起動している
- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

サーバテンプレートの格納先とファイル名

サーバテンプレートは、初回のドメイン起動後に展開されます。

サーバテンプレートのファイル名を次に示します。

- `httpsd.conf@linux.vtl`（Linux の場合）
  Web サーバの基本設定用サーバテンプレートです。リクエスト転送と負荷分散以外の基本設定を記述します。
- `httpsd.conf@aix.vtl`（AIX の場合）
  Web サーバの基本設定用サーバテンプレートです。リクエスト転送と負荷分散以外の基本設定を記述します。
- `reverse_proxy.conf@.vtl`
  Web サーバのリクエスト転送設定用サーバテンプレートです。リクエスト転送先サーバインスタンスがクラスター構成でない場合に、設定を記述します。
- `proxy_balancer.conf@.vtl`
  Web サーバの負荷分散設定用サーバテンプレートです。リクエスト転送先サーバインスタンスがクラスター構成の場合に、設定を記述します。

初回のドメイン起動時、サーバテンプレートの各ファイルは、*Application Serverのインストールディレクトリー*/javaee/glassfish/domains/*ドメイン名*/server_templates/webserver/conf に展開されます。

サーバテンプレートの編集方法

サーバテンプレートは、次の方法で編集できます。

- 拡張プロパティーを設定するために、VTL 構文を記述する
  サーバテンプレートに VTL 構文を記述し、asadmin ユーティリティーコマンドの set サブコマンドの拡張プロパティーの値を操作することで、Web サーバを設定します。
- ディレクティブを直接記述する
  サーバテンプレートにディレクティブを直接記述することで、Web サーバを設定します。

拡張プロパティーを設定するために、VTL 構文を記述すると、asadmin ユーティリティーコマンドのset サブコマンドで Web サーバの設定を変更し、get サブコマンドで変更内容を確認できるようになります。

サーバテンプレートは、asadmin ユーティリティーコマンドのcreate-webserver サブコマンド、および start-webserver サブコマンド実行時にドメイン管理サーバによって読み込まれ、Web サーバが読み込む定義ファイルに設定が反映されます。

操作手順

1. 拡張プロパティーを設定するために、VTL 構文を記述する場合は、次の手順で設定します。
   reverse_proxy.conf にProxyPreserveHost ディレクティブを設定する例で、手順を説明します。
   a. テキストエディターなどを利用して、サーバテンプレートのファイル（reverse_proxy.conf@.vtl）を開いて、拡張プロパティーの VTL 構文を記述します。
      拡張プロパティーは、接頭辞にex_を付加した名称で設定します。

      ProxyPreserveHost ディレクティブの場合、拡張プロパティーはex_ProxyPreserveHost.value で設定します。

      ```
      ProxyPreserveHost ${property.ex_ProxyPreserveHost.value}
      ```

   b. サーバテンプレートのファイルを保存します。
   c. asadmin ユーティリティーコマンドのset サブコマンドを実行して、Web サーバのコンフィグに対して、拡張プロパティー ex_ProxyPreserveHost.value の値にOn を設定します。

      ```
      asadmin set hitachi-webservers.hitachi-webserver.Webサーバ
      名.property.ex_ProxyPreserveHost=On
      ```

      コマンドの実行結果を次に示します。

      ```
      hitachi-webservers.hitachi-webserver.Webサーバ名.property.ex_ProxyPreserveHost=On
      Command set executed successfully.
      ```

d. asadmin ユーティリティーコマンドのget サブコマンドを実行して、拡張プロパティー ex_ProxyPreserveHost.value の値が設定したとおりになっているかを確認します。

```
asadmin get hitachi-webservers.hitachi-webserver.Webサーバ
名.property.ex_ProxyPreserveHost
```

コマンドの実行結果を次に示します。

```
hitachi-webservers.hitachi-webserver.Webサーバ名.property.ex_ProxyPreserveHost=On
Command get executed successfully.
```

2. ディレクティブを直接記述する場合は、次の手順で直接記述します。

a. テキストエディターなどを利用して、サーバテンプレートのファイルを開いて、Web Server のディレクティブを直接記述します。

ディレクティブをコメントとする場合は、先頭の#の直後に半角スペースを入れて記述します。

> **重要**
>
> サーバテンプレート上で、各種ログの出力先やファイル名を変更しないでください。システム情報収集機能での一括収集の対象外となります。ログの出力先はHJES_LOGSDIR 環境変数で変更できます。ファイル名は変更できません。

b. サーバテンプレートのファイルを保存します。

Web サーバの初期設定値

各サーバテンプレートのファイルから Web サーバを構築した場合にドメイン管理サーバによって読み込まれる、Web サーバの各定義ファイルの初期設定値を次に示します。

httpsd.conf の場合

```
Listen 80

StartServers 20
MinSpareServers 10
MaxSpareServers 20
MaxRequestWorkers 150
MaxConnectionsPerChild 10000
Timeout 30
KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 3
HostnameLookups Off

User bin
Group bin


ServerRoot "Application Serverのインストールディレクトリー/httpsd"

ServerName www.example.com
DocumentRoot "Application Serverのインストールディレクトリー/javaee/glassfish/nodes/
localhost-domain1/Web1/root/htdocs"
```

```
DirectoryIndex index.html
UseCanonicalName Off
ServerSignature Off
ServerTokens ProductOnly
TraceEnable Off

LogLevel warn
ErrorLog "|Application Serverのインストールディレクトリー/httpsd/sbin/rotatelogs
Application Serverのインストールディレクトリー/javaee/logs/nodes/localhost-domain1/
Web1/error 86400 -fnum 8 -diff 540"
HWSRequestLog "|Application Serverのインストールディレクトリー/httpsd/sbin/rotatelogs
Application Serverのインストールディレクトリー/javaee/logs/nodes/localhost-domain1/
Web1/hwsrequest 86400 -fnum 8 -diff 540"
LogFormat "%h %l %u %t ¥"%r¥" %>s %b ¥"%{Referer}i¥" ¥"%{User-Agent}i¥" %I %O"
combinedio
LogFormat "%h %l %u %t ¥"%r¥" %>s %b ¥"%{Referer}i¥" ¥"%{User-Agent}i¥"" combined
LogFormat "%h %l %u %t ¥"%r¥" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
LogFormat "%h %l %u %t ¥"%r¥" %>s %b %P %{hws_ap_root}n %I %O %X %D ¥"%{Referer}i¥"
¥"%{User-Agent}i¥"" hws_trace
LogFormat "%h %l %u %t ¥"%r¥" %>s %b %T %P %{hws_ap_root}n" hws_std
HWSLogTimeVerbose On
CustomLog "|Application Serverのインストールディレクトリー/httpsd/sbin/rotatelogs
Application Serverのインストールディレクトリー/javaee/logs/nodes/localhost-domain1/
Web1/access 86400 -fnum 8 -diff 540" hws_std
PidFile "Application Serverのインストールディレクトリー/javaee/logs/nodes/localhost-
domain1/Web1/httpd.pid"
HWSTraceIdFile "Application Serverのインストールディレクトリー/javaee/logs/nodes/
localhost-domain1/Web1/hws.trcid"
HWSTraceLogFile "Application Serverのインストールディレクトリー/javaee/logs/nodes/
localhost-domain1/Web1/hws.trclog"

SSLDisable

TypesConfig "Application Serverのインストールディレクトリー/httpsd/conf/mime.types"
AddEncoding x-compress .Z
AddEncoding x-gzip .gz .tgz
AddLanguage ca .ca
AddLanguage cs .cz .cs
AddLanguage da .dk
AddLanguage de .de
AddLanguage el .el
AddLanguage en .en
AddLanguage eo .eo
AddLanguage es .es
AddLanguage et .et
AddLanguage fr .fr
AddLanguage he .he
AddLanguage hr .hr
AddLanguage it .it
AddLanguage ja .ja
AddLanguage ko .ko
AddLanguage ltz .ltz
AddLanguage nl .nl
AddLanguage nn .nn
AddLanguage no .no
AddLanguage pl .po
```

```
AddLanguage pt .pt
AddLanguage pt-BR .pt-br
AddLanguage ru .ru
AddLanguage sv .sv
AddLanguage tr .tr
AddLanguage zh-CN .zh-cn
AddLanguage zh-TW .zh-tw

BrowserMatch "Mozilla/2" nokeepalive
BrowserMatch "MSIE 4¥.0b2;" nokeepalive downgrade-1.0 force-response-1.0
BrowserMatch "RealPlayer 4¥.0" force-response-1.0
BrowserMatch "Java/1¥.0" force-response-1.0
BrowserMatch "JDK/1¥.0" force-response-1.0
BrowserMatch "Microsoft Data Access Internet Publishing Provider" redirect-carefully
BrowserMatch "MS FrontPage" redirect-carefully
BrowserMatch "^WebDrive" redirect-carefully
BrowserMatch "^WebDAVFS/1.[01234]" redirect-carefully
BrowserMatch "^gnome-vfs/1.0" redirect-carefully
BrowserMatch "^XML Spy" redirect-carefully
BrowserMatch "^Dreamweaver-WebDAV-SCM1" redirect-carefully
BrowserMatch " Konqueror/4" redirect-carefully

Alias /icons/ "Application Serverのインストールディレクトリー/httpsd/icons/"
IndexOptions FancyIndexing
AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip
AddIconByType (TXT,/icons/text.gif) text/*
AddIconByType (IMG,/icons/image2.gif) image/*
AddIconByType (SND,/icons/sound2.gif) audio/*
AddIconByType (VID,/icons/movie.gif) video/*
AddIcon /icons/binary.gif .bin .exe
AddIcon /icons/binhex.gif .hqx
AddIcon /icons/tar.gif .tar
AddIcon /icons/world2.gif .wrl .wrl.gz .vrml .vrm .iv
AddIcon /icons/compressed.gif .Z .z .tgz .gz .zip
AddIcon /icons/a.gif .ps .ai .eps
AddIcon /icons/layout.gif .html .shtml .htm .pdf
AddIcon /icons/text.gif .txt
AddIcon /icons/c.gif .c
AddIcon /icons/p.gif .pl .py
AddIcon /icons/f.gif .for
AddIcon /icons/dvi.gif .dvi
AddIcon /icons/uuencoded.gif .uu
AddIcon /icons/script.gif .conf .sh .shar .csh .ksh .tcl
AddIcon /icons/tex.gif .tex
AddIcon /icons/bomb.gif core
AddIcon /icons/back.gif ..
AddIcon /icons/hand.right.gif README
AddIcon /icons/folder.gif ^^DIRECTORY^^
AddIcon /icons/blank.gif ^^BLANKICON^^
DefaultIcon /icons/unknown.gif
ReadmeName README.html
HeaderName HEADER.html
IndexIgnore .??* *~ *# HEADER* README* RCS CVS *,v *,t

<Directory />
        Options None
        AllowOverride None
</Directory>
```

```
<Directory "Application Serverのインストールディレクトリー/httpsd/htdocs">
        Options None
        AllowOverride None
</Directory>

<FilesMatch "^¥.(ht|key)">
        Order allow,deny
        Deny from all
</FilesMatch>

Include "Application Serverのインストールディレクトリー/javaee/glassfish/nodes/
localhost-domain1/Web1/root/conf/reverse_proxy.conf"

HWSGracefulStopLog On
HWSGracefulStopTimeout 0

HWSPrfId PRF1
```

reverse_proxy.conf の場合

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_http_module modules/mod_proxy_http.so
HWSSuppressModuleTrace mod_proxy.c hook
ProxyVia Off
ProxyTimeout 200
ProxyPass / http://ホスト名:28080/  connectiontimeout=2
ProxyPassReverse / http://ホスト名:28080/
HWSProxyPassReverseCookie /
```

proxy_balancer.conf の場合

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_http_module modules/mod_proxy_http.so
LoadModule proxy_balancer_module modules/mod_proxy_balancer.so
LoadModule lbmethod_byrequests_module modules/mod_lbmethod_byrequests.so
LoadModule slotmem_shm_module modules/mod_slotmem_shm.so
HWSSuppressModuleTrace mod_proxy.c hook
HWSSuppressModuleTrace mod_proxy_balancer.c
ProxyVia Off
ProxyTimeout 200
```

4.6.5 create-jvm-options サブコマンドを使用して JavaVM オプションを変更する

JavaVM オプションは、サーバインスタンスと、ドメイン管理サーバに対して指定できます。すでに指定されている JavaVM オプションを変更するには、asadmin ユーティリティーコマンドのdelete-jvm-options サブコマンドで削除してから、create-jvm-options サブコマンドで変更後の JavaVM オプションを指定します。

前提条件

- ドメイン管理サーバが起動している
- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのlist-jvm-options サブコマンドを実行して、クラスター内のすべてのサーバインスタンスのオプションの一覧を表示します。

```
asadmin list-jvm-options --target サーバインスタンス名またはクラスター名
```

   - 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

2. 手順 1 で表示した一覧に変更するオプションがある場合は、asadmin ユーティリティーコマンドの delete-jvm-options サブコマンドを実行して、変更するオプションを削除します。

```
asadmin delete-jvm-options --target サーバインスタンス名またはクラスター名
[オプション名[=オプションの値][:オプション名[=オプションの値]]...]
```

   - 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。
   - オプションを複数指定する場合は、コロン（:）で区切って指定します。

> **重要**
>
> オプション入力時、記号に対するエスケープ処理が必要になります。
>
> 例えば、-XX:MaxMetaspaceSize=192m を指定する場合は、:に対して¥¥を付けてエスケープ処理し、-XX¥¥:MaxMetaspaceSize=192m と入力します。

コマンドの実行結果を次に示します。

```
Deleted n option(s)
Command delete-jvm-options executed successfully.
```

n には、指定したオプションの数に応じて数字が埋め込まれます。

3. asadmin ユーティリティーコマンドのcreate-jvm-options サブコマンドを実行して、すべてのサーバインスタンスに対して、オプションで Java ヒープなどの Java メモリーの値を設定します。

```
asadmin create-jvm-options --target サーバインスタンス名またはクラスター名
[オプション名[=オプションの値][:オプション名[=オプションの値]]...]
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。
- オプションには、JavaVM のデフォルト値と、Java EE Server の初期値とが異なるものがあります。サーバインスタンスで JavaVM のデフォルト値と初期値が異なるオプションを次に示します。

表 4-2　サーバインスタンスで JavaVM のデフォルト値と初期値が異なるオプション

オプション名	Java EE Server の初期値
-XX:HitachiExplicitHeapMaxSize	-XX:HitachiExplicitHeapMaxSize=512m
-XX:SurvivorRatio	Linux の場合 -XX:SurvivorRatio=8
-XX:[+|-]HitachiUseExplicitMemory	-XX:+HitachiUseExplicitMemory
-XX:MaxMetaspaceSize	-XX:MaxMetaspaceSize=256m
-XX:MetaspaceSize	-XX:MetaspaceSize=256m
-Xms	-Xms1536m
-Xmx	-Xmx1536m

重要

- すでに指定されているオプションを変更する場合は、delete-jvm-options サブコマンドで変更するオプションを削除してください。削除しないまま、create-jvm-options サブコマンドでオプションを指定した場合は、警告が表示され、同じオプションが複数登録されます。警告が表示された場合は、list-jvm-options サブコマンドで指定されているオプションを確認し、不要なオプションを削除してください。
- ドメイン管理サーバは 1 時間ごと、サーバインスタンスは 24 時間ごとに SystemGC を実行します。SystemGC の実行間隔は、sun.rmi.dgc.server.gcInterval と、sun.rmi.dgc.client.gcInterval で変更できます。これらのシステムプロパティーで FullGC の発生間隔を広げても GC 発生回数が削減されない場合、Java ヒープが不足していることがあります。この場合は、Java ヒープのチューニングを実施することで改善することがあります。

コマンドの実行結果を次に示します。

```
Created n option(s)
Command create-jvm-options executed successfully.
```

n には、指定したオプションの数に応じて数字が埋め込まれます。

4. asadmin ユーティリティーコマンドのlist-jvm-options サブコマンドを実行して、すべてのサーバインスタンスのオプションの一覧を表示します。

```
asadmin list-jvm-options --target サーバインスタンス名またはクラスター名
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

手順 3 で指定したオプションの値が変更されていることを確認してください。

5. asadmin ユーティリティーコマンドのlist-jvm-options サブコマンドを実行して、ドメイン管理サーバのオプションの一覧を表示します。

```
asadmin list-jvm-options
```

6. 手順 5 で表示した一覧に変更するオプションがある場合は、asadmin ユーティリティーコマンドの delete-jvm-options サブコマンドを実行して、変更するオプションを削除します。

```
asadmin delete-jvm-options [オプション名[=オプションの値]
[:オプション名[=オプションの値]]...]
```

コマンドの実行結果を次に示します。

```
Deleted n option(s)
Command delete-jvm-options executed successfully.
```

n には、指定したオプションの数に応じて数字が埋め込まれます。

7. asadmin ユーティリティーコマンドのcreate-jvm-options サブコマンドを実行して、ドメイン管理サーバに対して、オプションで Java ヒープなどの Java メモリーの値を設定します。

```
asadmin create-jvm-options [オプション名[=オプションの値]
[:オプション名[=オプションの値]]...]
```

オプション名[=*オプションの値*]には、オプションとして、-Xms1024m、-Xmx1024m などを設定します。

オプションには、JavaVM のデフォルト値と、Java EE Server の初期値とが異なるものがあります。ドメイン管理サーバで JavaVM のデフォルト値と初期値が異なるオプションを次に示します。

表 4-3　ドメイン管理サーバで JavaVM のデフォルト値と初期値が異なるオプション

オプション名	Java EE Server の初期値
-XX:MaxMetaspaceSize	-XX:MaxMetaspaceSize=192m
-XX:MetaspaceSize	-XX:MetaspaceSize=192m
-Xms	-Xms512m

オプション名	Java EE Server の初期値
-Xmx	-Xmx512m

重要

ドメイン管理サーバの Java ヒープサイズは、デプロイするアプリケーションのアーカイブのサイズに応じて調整してください。アプリケーションのアーカイブのサイズによっては、ドメイン管理サーバの Java ヒープサイズが枯渇し、メモリー不足となることがあります。

また、ドメイン管理サーバの Java ヒープサイズに不適切な値（極端に小さい値や大きい値など）を指定した場合、ドメイン管理サーバが起動しなくなり、ドメインの再構築が必要になることがあります。

このような事態を避けるために、ドメイン管理サーバのオプションを変更する場合は、あらかじめ、backup-domain コマンドを実行して、ドメインのバックアップを取得しておくことをお勧めします。

コマンドの実行結果を次に示します。

```
Created n option(s)
Command create-jvm-options executed successfully.
```

n には、指定したオプションの数に応じて数字が埋め込まれます。

8. asadmin ユーティリティーコマンドのlist-jvm-options サブコマンドを実行して、ドメイン管理サーバのオプションの一覧を表示します。

```
asadmin list-jvm-options
```

手順 7 で指定したオプションの値が変更されていることを確認してください。

9. サーバインスタンス、およびドメイン管理サーバに対して、Java メモリー以外のオプションを指定する場合は、手順 1 から手順 8 までを繰り返します。

オプションには、JavaVM のデフォルト値と、Java EE Server の初期値とが異なるものがあります。JavaVM のデフォルト値と初期値が異なるオプションを次に示します。

表 4-4　JavaVM のデフォルト値と初期値が異なるオプション

分類	オプション名	Java EE Server の初期値
サーバインスタンスの場合	-XX:HitachiExplicitMemoryJavaLog	-XX:HitachiExplicitMemoryJavaLog:*Java EE Server のインストールディレクトリー*/logs/nodes/*ノード名*/*サーバインスタンス名*/je_eheap_event
	-XX:HitachiExplicitMemoryLogLevel	-XX:HitachiExplicitMemoryLogLevel:normal

分類	オプション名	Java EE Server の初期値
	-XX:HitachiJavaLog	-XX:HitachiJavaLog:*Java EE Serverのインストールディレクトリー*/logs/nodes/*ノード名*/*サーバインスタンス名*/je_javavm
	-XX:[+|-]HitachiOutOfMemoryCause	-XX:+HitachiOutOfMemoryCause
	-XX:[+|-]HitachiOutOfMemorySize	-XX:+HitachiOutOfMemorySize
	-XX:[+|-]HitachiTrueTypeInLocals	-XX:+HitachiTrueTypeInLocals
	-XX:[+|-]UseCompressedOops	Linux（AMD64 & Intel EM64T）の場合： -XX:+UseCompressedOops
	-Xhras（指定なし）	-Xhras
ドメイン管理サーバの場合	-XX:HitachiJavaLog	-XX:HitachiJavaLog:*Java EE Serverのインストールディレクトリー*/logs/domains/*ドメイン名*/das_javavm
	-XX:[+|-]HitachiOutOfMemoryCause	-XX:+HitachiOutOfMemoryCause
	-XX:[+|-]HitachiOutOfMemorySize	-XX:+HitachiOutOfMemorySize
	-XX:[+|-]HitachiTrueTypeInLocals	-XX:+HitachiTrueTypeInLocals
	-XX:[+|-]UseCompressedOops	Linux（AMD64 & Intel EM64T）の場合： -XX:+UseCompressedOops
	-Xhras（指定なし）	-Xhras

4.6.6　トラブルシュート資料を一括収集するための設定をする

障害の原因究明に必要なトラブルシュート資料は、システム情報収集機能を使用して一括収集できます。システム情報収集機能を使用してトラブルシュート資料を一括収集するためには、トラブルシュート資料を収集するコマンドのスクリプトファイルを編集します。障害発生時には、このコマンドが自動で実行されて、トラブルシュート資料が収集されます。なお、ドメインのデフォルトの管理ユーザーおよびパスワードを変更していない場合は、編集する必要はありません。

前提条件

- ドメイン管理サーバが起動している
- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

操作手順

1. トラブルシュート資料を一括収集するコマンドのスクリプトファイルを編集します。

   ドメインのデフォルトの管理ユーザーおよびパスワードを変更していない場合は、編集する必要はありません。

   編集するスクリプトファイル

   Java EE Serverのインストールディレクトリー/glassfish/config/manager/snapshot_event-hook

   編集する内容

   スクリプトファイルをテキストエディタで開いて、次の設定値を編集します。

   ```
   ADMIN_HOST=ドメイン管理サーバのホスト名
   USER_ID=ドメイン管理サーバのユーザー名
   PWDFILE=パスワードファイルのパス※
   ```

   注※ AS_ADMIN_PASSWORD=*ドメイン管理サーバのパスワード*を記述したパスワードファイルのファイルパスを指定してください。パスワードを設定していない場合は、PWDFILE の値に空文字を指定してください。

   （例）パスワードを設定していない場合

   ```
   PWDFILE=
   ```

関連項目

- 9.1　Application Server が出力するトラブルシュート資料について

4.6.7　ポート番号を変更する

ポート番号を変更するには、asadmin ユーティリティーコマンドのset サブコマンドで、ポート関連パラメーターに変更後のポート番号を設定します。

前提条件

- ドメイン管理サーバが起動している
- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

ポート番号の変更で指定するポート関連パラメーター

ポート番号を変更するときに、asadmin ユーティリティーコマンドのget サブコマンド、およびset サブコマンドで指定するポート関連パラメーターを次に示します。

- configs.config.*Java EEサーバのコンフィグ名*.iiop-service.iiop-listener.*ID*.port
- configs.config.*Java EEサーバのコンフィグ名*.network-config.network-listeners.network-listener.*リスナー名*.port
- configs.config.*Java EEサーバのコンフィグ名*.admin-service.jmx-connector.system.port
- configs.config.*Java EEサーバのコンフィグ名*.jms-service.jms-host.*JMSホスト名*.port
- hitachi-webservers.hitachi-webserver.*Webサーバ名*.property.listen-port
- hitachi-webservers.hitachi-webserver.*Webサーバ名*.property.listen-add-port*n*
- hitachi-webservers.hitachi-webserver.*Webサーバ名*.property.server-name
- hitachi-webserver-configs.hitachi-webserver-config.*Webサーバのコンフィグ名*.property.listen-port
- hitachi-webserver-configs.hitachi-webserver-config.*Webサーバのコンフィグ名*.property.listen-add-port*n*
- hitachi-webserver-configs.hitachi-webserver-config.*Webサーバのコンフィグ名*.property.server-name

操作手順

1. asadmin ユーティリティーコマンドのget サブコマンドを実行して、ポート関連パラメーターの設定値を取得して、変更前のポート番号を確認します。

```
asadmin get ポート関連パラメーター名
```

コマンドの実行結果を次に示します。

```
Command get executed successfully.
```

2. asadmin ユーティリティーコマンドのset サブコマンドを実行して、変更後のポート番号を設定します。

```
asadmin set ポート関連パラメーター名=変更後のポート番号
```

コマンドの実行結果を次に示します。

```
Command set executed successfully.
```

4.6.8 Java デバッガー用通信ポート番号を変更する

Java デバッガー用通信ポート番号を変更するには、asadmin ユーティリティーコマンドのdelete-system-property サブコマンドで変更前のポート番号を削除してから、create-system-properties サブコマンドで変更後のポート番号を設定します。

前提条件

- ドメイン管理サーバが起動している
- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのlist-system-properties サブコマンドを実行して、Java デバッガー用通信ポート番号のシステムプロパティー（JAVA_DEBUGGER_PORT）の設定値を取得して、変更前のポート番号を確認します。

```
asadmin list-system-properties サーバインスタンス名またはクラスター名
```

- 1 つの Java EE サーバを配置する構成の場合は、サーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、クラスター名を指定します。

コマンドの実行結果を次に示します。

```
JAVA_DEBUGGER_PORT=29009
    :
Command list-system-properties executed successfully.
```

> **メモ**
>
> システムプロパティー（JAVA_DEBUGGER_PORT）が設定されている場合には、手順 2、および手順 3 を実行してください。設定されていない場合には、手順 3 を実行してください。

2. asadmin ユーティリティーコマンドのdelete-system-property サブコマンドを実行して、システムプロパティーJAVA_DEBUGGER_PORT を削除します。

```
asadmin delete-system-property --target サーバインスタンス名またはクラスター名
JAVA_DEBUGGER_PORT
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

コマンドの実行結果を次に示します。

```
Command delete-system-property executed successfully.
```

3. asadmin ユーティリティーコマンドのcreate-system-properties サブコマンドを実行して、システムプロパティーJAVA_DEBUGGER_PORT に変更後のポート番号を設定します。

```
asadmin create-system-properties --target サーバインスタンス名またはクラスター名
JAVA_DEBUGGER_PORT=変更後のポート番号
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

コマンドの実行結果を次に示します。

```
Command create-system-properties executed successfully.
```

4.6.9 ポートをオープンする

クローズしているポートをオープンするには、asadmin ユーティリティーコマンドのset サブコマンドで、ポート関連パラメーターの設定を有効にします。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している

想定ユーザー

- システム構築者

ポートのオープンで指定するポート関連パラメーター

ポートをオープンするときに、asadmin ユーティリティーコマンドのget サブコマンド、およびset サブコマンドで指定するポート関連パラメーターを次に示します。

- configs.config.*Java EEサーバのコンフィグ名*.network-config.network-listeners.network-listener.*リスナー名*.enabled

  このパラメーターでは、configs.config.*Java EEサーバのコンフィグ名*.network-config.network-listeners.network-listener.*リスナー名*.port でポート番号を指定したポートをオープンします。

操作手順

1. asadmin ユーティリティーコマンドのget サブコマンドを実行して、ポート関連パラメーターの設定値を取得して、ポートがクローズしていること（設定値がfalse であること）を確認します。

```
asadmin get ポート関連パラメーター名
```

コマンドの実行結果を次に示します。

```
Command get executed successfully.
```

2. asadmin ユーティリティーコマンドのset サブコマンドを実行して、クローズしているポートをオープンします。

```
asadmin set ポート関連パラメーター名=true
```

ポート関連パラメーターの値にtrue を指定し、ポート関連パラメーターの設定を有効にすることで、ポートをオープンします。

コマンドの実行結果を次に示します。

```
Command set executed successfully.
```

4.6.10 IIOP リスナーのポートをオープンする

IIOP リスナーのポートをオープンするには、asadmin ユーティリティーコマンドのcreate-iiop-listener サブコマンドで IIOP リスナーを作成します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している
- 使用できるポート番号を確認済みである

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのcreate-iiop-listener サブコマンドを実行して、IIOP リスナーを作成してポートをオープンします。

```
asadmin create-iiop-listener --listeneraddress 0.0.0.0 --iiopport ポート番号
--target サーバインスタンス名またはクラスター名 IIOPリスナー名
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

コマンドの実行結果を次に示します。

```
Command create-iiop-listener executed successfully.
```

4.6.11　JMS ホストのポートをオープンする

JMS ホストのポートをオープンするには、asadmin ユーティリティーコマンドのcreate-jms-host サブコマンドで JMS ホストを作成します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している
- 使用できるポート番号を確認済みである

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのcreate-jms-host サブコマンドを実行して、JMS ホストを作成してポートをオープンします。

```
asadmin create-jms-host --mqhost JMSサービスのホスト名 --mqport JMSサービスのポート番号
--mquser JMSサービスのユーザー名 --mqpassword JMSサービスのパスワード
--target サーバインスタンス名またはクラスター名 JMSホスト名
```

   - 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

   コマンドの実行結果を次に示します。

```
Command create-jms-host executed successfully.
```

4.6.12　ポートをクローズする

オープンしているポートをクローズするには、asadmin ユーティリティーコマンドのset サブコマンドで、ポート関連パラメーターの設定を無効にします。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している

想定ユーザー

- システム構築者

ポートのクローズで指定するポート関連パラメーター

ポートをクローズするときに、asadmin ユーティリティーコマンドのget サブコマンド、およびset サブコマンドで指定するポート関連パラメーターを次に示します。

- configs.config.*Java EEサーバのコンフィグ名*.network-config.network-listeners.network-listener.*リスナー名*.enabled

  このパラメーターでは、configs.config.*Java EEサーバのコンフィグ名*.network-config.network-listeners.network-listener.*リスナー名*.port でポート番号を指定したポートをクローズします。

操作手順

1. asadmin ユーティリティーコマンドのget サブコマンドを実行して、ポート関連パラメーターの設定値を取得して、ポートがオープンしていること（設定値がtrue であること）を確認します。

   ```
   asadmin get ポート関連パラメーター名
   ```

   コマンドの実行結果を次に示します。

   ```
   Command get executed successfully.
   ```

2. asadmin ユーティリティーコマンドのset サブコマンドを実行して、オープンしているポートをクローズします。

   ```
   asadmin set ポート関連パラメーター名=false
   ```

   ポート関連パラメーターの値にfalse を指定して、ポート関連パラメーターの設定を無効にすることで、ポートをクローズします。

   コマンドの実行結果を次に示します。

   ```
   Command set executed successfully.
   ```

4.6.13　IIOP リスナーのポートをクローズする

IIOP リスナーのポートをクローズするには、asadmin ユーティリティーコマンドのdelete-iiop-listener サブコマンドで IIOP リスナーを削除します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのlist-iiop-listeners サブコマンドを実行して、ポートをクローズする IIOP リスナーを確認します。

```
asadmin list-iiop-listeners サーバインスタンス名またはクラスター名
```

- 1 つの Java EE サーバを配置する構成の場合は、サーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、クラスター名を指定します。

コマンドの実行結果を次に示します。

```
Command list-iiop-listeners executed successfully.
```

2. asadmin ユーティリティーコマンドのdelete-iiop-listener サブコマンドを実行して、IIOP リスナーを削除してポートをクローズします。

```
asadmin delete-iiop-listener --target サーバインスタンス名またはクラスター名 IIOPリスナー名
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

コマンドの実行結果を次に示します。

```
Command delete-iiop-listener executed successfully.
```

4.6.14　JMS ホストのポートをクローズする

JMS ホストのポートをクローズするには、asadmin ユーティリティーコマンドのdelete-jms-host サブコマンドで JMS ホストを削除します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのlist-jms-hosts サブコマンドを実行して、ポートをクローズするJMS ホストを確認します。

```
asadmin list-jms-hosts --target サーバインスタンス名またはクラスター名
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

コマンドの実行結果を次に示します。

```
Command list-jms-hosts executed successfully.
```

2. asadmin ユーティリティーコマンドのdelete-jms-host サブコマンドを実行して、JMS ホストを削除してポートをクローズします。

```
asadmin delete-jms-host --target サーバインスタンス名またはクラスター名 JMSホスト名
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

コマンドの実行結果を次に示します。

```
Command delete-jms-host executed successfully.
```

4.6.15 asadmin ユーティリティーコマンドのプロセスに適用する環境変数を変更する

asadmin ユーティリティーコマンドのプロセスに適用する環境変数を変更するには、Java EE Server の環境変数定義ファイル（asenv.conf）を編集します。asadmin ユーティリティーコマンドのプロセスに適用する環境変数には、Java ヒープなどの Java メモリーの値や、asadmin ユーティリティーコマンドのログなどを設定します。例えば、デプロイしたアプリケーション数やアプリケーション中のファイル数が多く、Java EE サーバの起動時にメモリー不足になる場合は、asadmin ユーティリティーコマンドのプロセスに適用する Java ヒープ領域のサイズを変更します。

前提条件

- ドメイン管理サーバが起動している
- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

操作手順

1. **Java EE Server の環境変数定義ファイル（asenv.conf）を編集して、asadmin ユーティリティーコマンドのプロセスに適用する環境変数を変更します。**

   複数の Java EE サーバを配置するクラスター構成の場合は、リモートホストおよびローカルホストのそれぞれに対して、環境変数を変更します。

   Java EE Server の環境変数定義ファイル

   Application Serverのインストールディレクトリー/javaee/glassfish/config/asenv.conf

   編集の例

   例えば、Java ヒープ領域の最大サイズを変更する場合は、環境変数 HJES_ASADMIN_JVM_OPTIONS に値を指定します。

   ```
   HJES_ASADMIN_JVM_OPTIONS=-Xmx256m
   ```

4.6.16　Application Server に JDBC ドライバーをインストールする

Application Server に JDBC ドライバーをインストールするには、データベースのベンダーが提供する JDBC ドライバーの JAR ファイルを、Application Server にコピーします。JDBC ドライバーを有効にするために、asadmin ユーティリティーコマンドのrestart-domain サブコマンドでドメイン管理サーバを再起動します。

前提条件

- ドメイン管理サーバが起動している

想定ユーザー

- システム構築者

操作手順

1. **Application Server に JDBC ドライバーの JAR ファイルをコピーします。**

   JDBC ドライバーの JAR ファイルは、データベースのベンダーから提供されているものを使用します。コピー先ディレクトリーは、*Application Serverのインストールディレクトリー*/javaee/glassfish/domains/*ドメイン名*/lib です。

2. asadmin ユーティリティーコマンドのrestart-domain サブコマンドを実行して、ドメイン管理サーバを再起動します。

```
asadmin restart-domain
```

コマンドの実行結果を次に示します。

```
Command restart-domain executed successfully.
```

関連項目

- 4.7.2　データベースサーバへの接続を設定する

4.6.17　Application Server を起動する

Application Server を起動するには、asadmin ユーティリティーコマンドのstart-servers サブコマンドを実行します。

前提条件

- ドメイン管理サーバが起動している
- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのstart-servers サブコマンドを実行して、Application Server を一括で起動します。

```
asadmin start-servers
```

コマンドの実行結果を次に示します。

```
Command start-servers executed successfully.
```

2. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがrunning になっていることを確認してください。

```
パフォーマンストレーサー名 running
Command list-prfs executed successfully.
```

3. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがrunning になっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 running
Command list-instances executed successfully.
```

- クラスター名は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。
- サーバインスタンスの起動に成功したかどうかは、Java EE サーバ（サーバインスタンス）のメッセージログに KDKD20031-I が出力されているかどうか、または Administration Console の Java EE サーバ（サーバインスタンス）のステータスでも確認できます。

4. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがrunning になっていることを確認してください。

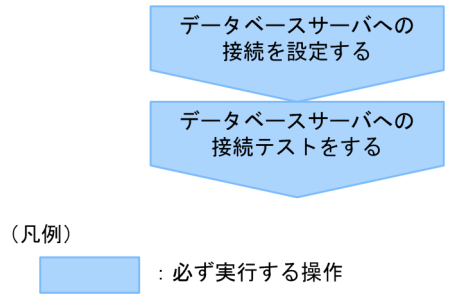
```
Webサーバ名    running
Command list-webservers executed successfully.
```

4.7　データベースサーバへの接続

Application Server とデータベースサーバを接続する手順について説明します。データベースサーバと接続するためには、Application Server にデータベースサーバと接続するための設定が必要です。

4.7.1　データベースサーバへの接続の流れ

Application Server とデータベースサーバを接続するために実施する作業と、作業の流れについて説明します。Application Server にデータベースサーバと接続するための設定をします。データベースサーバと接続できるかどうかを確認するために、接続テストを実施します。



関連項目

- 4.7.2　データベースサーバへの接続を設定する
- 4.7.3　データベースサーバへの接続テストをする

4.7.2　データベースサーバへの接続を設定する

データベースサーバへの接続を設定するには、asadmin ユーティリティーコマンドのcreate-jdbc-connection-pool サブコマンドで JDBC コネクションプールを作成し、create-jdbc-resource サブコマンドで JDBC リソースを作成します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している
- データベースサーバ（DBMS）が起動している
- Application Server で JDBC ドライバーのインストールが完了している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのcreate-jdbc-connection-pool サブコマンドを実行して、JDBC コネクションプールを作成します。

```
asadmin create-jdbc-connection-pool コネクションプールの設定
接続先ベンダー固有の設定 コネクションプールID
```

- *コネクションプールの設定*には、コネクションプールサイズや、コネクション障害検知機能の使用などをサブコマンドのオプションで設定します。
- *接続先ベンダー固有の設定*には、データベースのベンダーが提供しているドライバー固有情報をサブコマンドのオプションで設定します。
- *コネクションプール ID* には、JDBC コネクションプールを識別するための名称を設定します。

（例 1）Oracle の場合

```
asadmin create-jdbc-connection-pool
--datasourceclassname oracle.jdbc.pool.OracleDataSource
--restype javax.sql.DataSource
--property user=ユーザー名:password=パスワード:
url="jdbc¥:oracle¥:thin¥:@IPアドレス¥:ポート番号¥:Oracle SID"
コネクションプールID
```

（例 2）HiRDB の場合

```
asadmin create-jdbc-connection-pool
--datasourceclassname JP.co.Hitachi.soft.HiRDB.JDBC.PrdbDataSource
--restype javax.sql.DataSource
--property user=ユーザー名:password=パスワード:DBHostName=IPアドレス:
description=ポート番号 コネクションプールID
```

コマンドの実行結果を次に示します。

```
JDBC connection pool コネクションプールID created successfully.
Command create-jdbc-connection-pool executed successfully.
```

2. asadmin ユーティリティーコマンドのcreate-jdbc-resource サブコマンドを実行して、クラスター内のすべてのサーバインスタンスに対して JDBC リソースを作成します。

```
asadmin create-jdbc-resource --connectionpoolid コネクションプールID
--target サーバインスタンス名またはクラスター名 JNDI名
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。
- *JNDI 名*には、データベースサーバ（DBMS）の接続を取得するための名称を指定します。

コマンドの実行結果を次に示します。

```
JDBC resource JNDI名 created successfully.
Command create-jdbc-resource executed successfully.
```

次の作業

- データベースサーバへの接続テストをする

関連項目

- 4.7.3　データベースサーバへの接続テストをする

4.7.3　データベースサーバへの接続テストをする

データベースサーバへの接続テストをするには、asadmin ユーティリティーコマンドのping-connection-pool サブコマンドで、サーバインスタンスからデータベースサーバへ接続できていることを確認します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している
- データベースサーバ（DBMS）が起動している
- データベースサーバへ接続するための設定が完了している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのping-connection-pool サブコマンドを実行して、サーバインスタンスからデータベースサーバに接続できるかどうかを確認します。

```
asadmin ping-connection-pool --target サーバインスタンス名
コネクションプールID
```

コマンドの実行結果を次に示します。

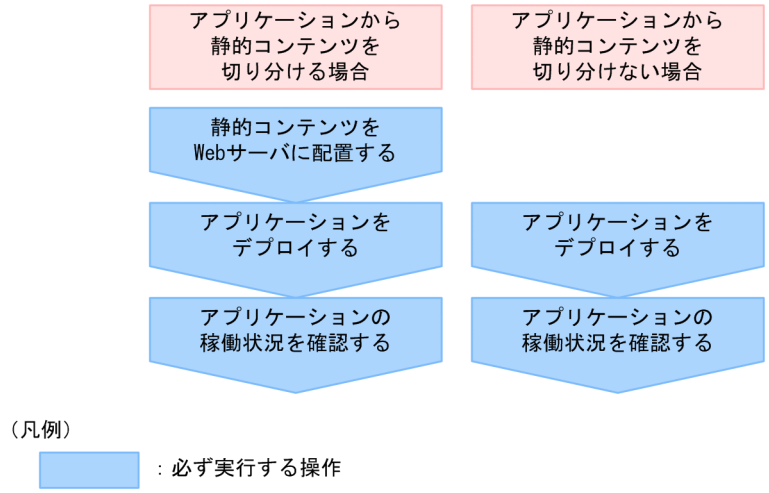
```
Command ping-connection-pool executed successfully.
```

4.8　アプリケーションのデプロイ

アプリケーションをデプロイする手順について説明します。アプリケーションから静的コンテンツを切り分けて、Web サーバに配置します。アプリケーション（動的コンテンツ）は、Java EE サーバ（サーバインスタンス）にデプロイします。デプロイすると、アプリケーションは開始されます。

4.8.1　アプリケーションのデプロイの流れ

アプリケーションをデプロイするために実施する作業と、作業の流れについて説明します。システムのパフォーマンスを向上させる場合は、アプリケーションから静的コンテンツを切り分けて Web サーバに配置します。アプリケーション（動的コンテンツ）は、Java EE サーバ（サーバインスタンス）にデプロイします。アプリケーションが開始しているかどうかを稼働状況で確認します。



関連項目

- 4.8.2　静的コンテンツを Web サーバに配置する
- 4.8.3　アプリケーションをデプロイする
- 4.8.4　アプリケーションの稼働状況を確認する

4.8.2　静的コンテンツを Web サーバに配置する

静的コンテンツを Web サーバに配置するには、Web サーバのドキュメントルートディレクトリーに格納します。Web サーバに静的コンテンツを配置し、ネットワークアクセスの回数や、やり取りするデータのサイズを減らすことで、パフォーマンスを向上できます。

前提条件

- ドメイン管理サーバが起動している

- Application Server が起動している
- アプリケーション開発者からアプリケーションを入手済みである

想定ユーザー

- システム構築者

操作手順

1. 静的コンテンツを Web サーバのドキュメントルートディレクトリーに格納します。
   Web サーバのドキュメントルートディレクトリーは、*Application Serverのインストールディレクトリー*/javaee/glassfish/nodes/*ノード名*/*Webサーバ名*/root/htdocs です。

次の作業

- アプリケーションをデプロイする

関連項目

- 4.8.3　アプリケーションをデプロイする

4.8.3　アプリケーションをデプロイする

アプリケーションをデプロイするには、asadmin ユーティリティーコマンドのdeploy サブコマンドを実行します。アプリケーションは、Java EE サーバ（サーバインスタンス）にデプロイすると開始されます。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している
- アプリケーション開発者からアプリケーションを入手済みである

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのdeploy サブコマンドを実行して、アプリケーションをサーバインスタンスにデプロイします。

```
asadmin deploy --target サーバインスタンス名またはクラスター名
アプリケーションのファイルパス
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

コマンドの実行結果を次に示します。

```
Application deployed with name アプリケーション名.
Command deploy executed successfully.
```

2. アプリケーションごとに、手順 1 を繰り返します。

注意事項

1. デプロイするアプリケーションの文字の範囲について

   次の項目に設定する文字には、英数字、ハイフン（-）、アンダースコア（_）、ピリオド（.）だけが使用できます。また、これらの文字列の先頭には、英数字またはアンダースコア（_）だけが使用できます。

   - deploy サブコマンドとredeploy サブコマンドに指定する、--name オプションのcomponent_name
   - deploy サブコマンドとredeploy サブコマンドに指定するアーカイブファイル名
   - DD のapplication-name 要素またはmodule-name 要素
   - クラスファイルなど、Java SE 仕様に準拠するファイル以外の、アプリケーション内のすべてのファイル名およびディレクトリー名

2. デプロイするアプリケーションに含めるモジュールのファイルパスについて

   デプロイするアプリケーションに含めるモジュール（WAR、RAR、EJB-JAR、アプリケーションクライアント JAR）のファイルパスは、次の置換規則に従って置換されるため、各モジュールのファイルパスが置換された結果がアプリケーション内で重複しないように、各モジュールのファイルパスを設定してください。

   - ファイルパスに含まれるディレクトリー区切り文字を”__”（アンダースコア 2 つ）に変換します。
   - 末尾が”.war”、”.jar”、”.rar”であるファイルパスに対して次の処理をします。

     末尾が”.war”の場合：「最初に現れる”.war”の手前までの文字列」と”_war”を結合

     末尾が”.jar”の場合：「最初に現れる”.jar”の手前までの文字列」と”_jar”を結合

     末尾が”.rar”の場合：「最初に現れる”.rar”の手前までの文字列」と”_rar”を結合

   （例 1）次のファイルパスは、置換された結果がすべて重複します。

   ファイルパス 1：*EARのトップディレクトリー*/war/war1_/sample-web.war

   ファイルパス 2：*EARのトップディレクトリー*/war/war1/_sample-web.war

   ファイルパス 3：*EARのトップディレクトリー*/war/war1___sample-web.war

   置換された結果：war__war1___sample-web_war

   （例 2）次のファイルパスは、置換された結果がすべて重複します。

   ファイルパス 1：*EARのトップディレクトリー*/war/sample-web.war

   ファイルパス 2：*EARのトップディレクトリー*/war/sample-web.war.war

ファイルパス 3：*EARのトップディレクトリー*/war/sample-web.war.web.war

置換された結果：war__sample-web_war

次の作業

- アプリケーションの稼働状況を確認する

関連項目

- 4.8.4　アプリケーションの稼働状況を確認する

4.8.4　アプリケーションの稼働状況を確認する

アプリケーションの稼働状況を確認するには、asadmin ユーティリティーコマンドのlist-applications サブコマンドでアプリケーションの一覧を表示します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している
- アプリケーションのデプロイが完了している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのlist-applications サブコマンドに--long オプションを指定して実行し、サーバインスタンスにデプロイしたアプリケーションの一覧を表示します。

```
asadmin list-applications --long=true サーバインスタンス名またはクラスター名
```

- 1 つの Java EE サーバを配置する構成の場合は、サーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、クラスター名を指定します。

コマンドの実行結果を次に示します。デプロイしたすべてのアプリケーションのステータスがenabled になっていることを確認してください。

```
NAME                    TYPE         STATUS
アプリケーション名  <ear, web>     enabled
アプリケーション名  <web>          enabled
Command list-applications executed successfully.
```

4.9　システム設定情報の確認

システム設定情報を確認する手順について説明します。システム設定情報とは、Application Server を構築したホストや、Application Server 内の各サーバの構成、各サーバの設定値などのことです。これらの情報は、コマンドで出力するファイル、または Administration Console で表示するペインで確認できます。

4.9.1　システム設定情報の確認について

システム設定情報とは、Application Server を構築したホストや、Application Server 内の各サーバの構成、各サーバの設定値などのことです。システム設定情報の種類と確認方法、および Application Server のセットアップ、データベースサーバへの接続、およびアプリケーションのデプロイを実施したあとのシステム構成について説明します。

システム設定情報の種類と確認方法

システム設定情報の種類には、次のものがあります。

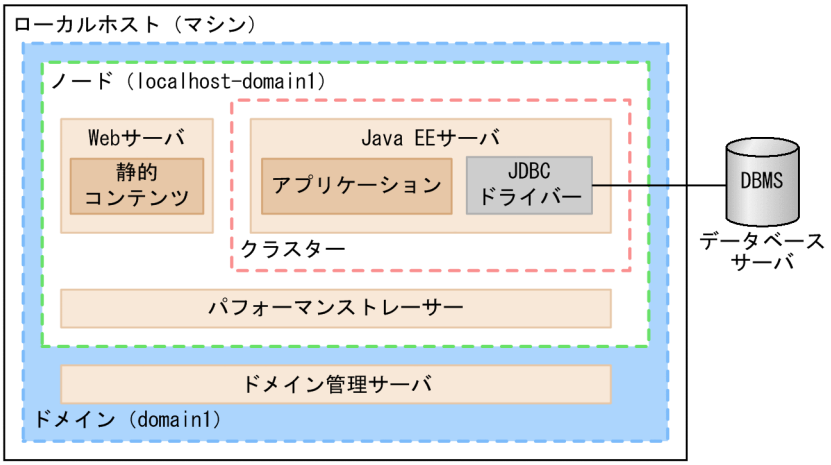
- サーバで設定しているシステムプロパティー、プロパティー、ファイルの定義項目などの名称とその設定値
- サーバの構成情報と稼働状況
- ドメイン管理サーバおよびサーバの構成情報と、名前、種別、ノードなどの基本情報

これらの情報は、コマンド、または Administration Console の画面を利用して確認できます。Web サーバの設定情報は、サーバテンプレートから確認します。コマンドの場合はファイルに出力することもできます。

アプリケーションのデプロイ後のシステム構成

Application Server のインストール後、次の作業を実施したあとのシステム構成を示します。

1. Application Server のセットアップ
2. データベースサーバへの接続
3. アプリケーションのデプロイ



4.9.2　コマンドを利用してシステム設定情報を確認する

コマンドを利用してシステム設定情報を確認するには、asadmin ユーティリティーコマンドの list-から始まるサブコマンドやget サブコマンドで一覧をファイルに出力します。ファイルの内容が、設定したとおりになっているかを確認します。すべての標準プロパティー、および拡張プロパティーの設定値は、get サブコマンドで参照できます。これらのプロパティー以外でテンプレートに設定した情報は、サーバテンプレートで確認してください。

前提条件

- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのlist-instances サブコマンドを実行して、ドメイン内のすべての Java EE サーバと、その詳細情報（ホスト名、ポート番号などの情報）の一覧をファイルに出力します。

```
asadmin list-instances --long=true
>> 出力ファイルのファイルパス
```

2. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、ドメイン内のすべてのパフォーマンストレーサーと、その詳細情報（ホスト名、プロセス ID などの情報）の一覧をファイルに出力します。

```
asadmin list-prfs --long=true
>> 出力ファイルのファイルパス
```

3. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、ドメイン内のすべての Web サーバと、その詳細情報（ホスト名、プロセス ID などの情報）の一覧をファイルに出力します。

```
asadmin list-webservers --long=true >> 出力ファイルのファイルパス
```

4. asadmin ユーティリティーコマンドのlist-clusters サブコマンドを実行して、ドメイン内のすべてのクラスターの一覧をファイルに出力します。

```
asadmin list-clusters >> 出力ファイルのファイルパス
```

5. asadmin ユーティリティーコマンドのlist-relations サブコマンドを実行して、ドメイン内のすべてのサーバ間関連と、その詳細情報（関連の種類、関連元、関連先などの情報）の一覧をファイルに出力します。

```
asadmin list-relations --long=true >> 出力ファイルのファイルパス
```

6. asadmin ユーティリティーコマンドの list-から始まるサブコマンドを実行して、設定されている情報の一覧をファイルに出力します。

（例）

list-jvm-options サブコマンド、list-system-properties サブコマンドを実行して、JavaVM オプションとシステムプロパティーの一覧をファイルに出力します。

```
asadmin list-jvm-options --target サーバインスタンス名またはクラスター名
>> 出力ファイルのファイルパス
asadmin list-jvm-options --target 構成名 >> 出力ファイルのファイルパス
asadmin list-system-properties サーバインスタンス名またはクラスター名
>> 出力ファイルのファイルパス
asadmin list-system-properties 構成名 >> 出力ファイルのファイルパス
```

- 1 つの Java EE サーバを配置する構成の場合は、サーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、クラスター名を指定します。

7. asadmin ユーティリティーコマンドのget サブコマンドを実行して、各サーバの設定内容をファイルに出力します。

```
asadmin get "*" >> 出力ファイルのファイルパス
```

8. 手順 1～手順 7 で出力したファイルの内容が、設定したとおりのシステム設定情報になっているかを確認します。

メモ

すべての標準プロパティー、および拡張プロパティーの設定値は、get サブコマンドで参照できます。ただし、サーバテンプレートを使用して設定した、標準プロパティーおよび拡張プロパティー以外の情報は、サーバテンプレートから確認してください。

4.9.3　サーバテンプレートから Web サーバの設定情報を確認する

Web サーバの設定情報は、サーバテンプレートのファイルの内容から、設定したとおりになっているかを確認します。

前提条件

- ドメイン管理サーバが起動している
- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのget サブコマンドを実行して、Web サーバのコンフィグ名を取得します。

```
asadmin get hitachi-webservers.hitachi-webserver.Webサーバ名.
hitachi-webserver-config-ref
```

2. 手順 1 で取得した Web サーバのコンフィグ名を指定し、asadmin ユーティリティーコマンドのget サブコマンドを実行して、サーバテンプレートのファイルパスを取得します。

```
asadmin get hitachi-webserver-configs.hitachi-webserver-config.
Webサーバのコンフィグ名.hitachi-manage-info.template-path
```

   取得したファイルパスに含まれる${com.sun.aas.instanceRoot} は、*Application Serverのインストールディレクトリー*/javaee/glassfish/domains/*ドメイン名*を示しています。

3. 手順 2 で取得したファイルパスにあるサーバテンプレートを開いて次のファイルの内容と比較し、設定したとおりになっているかを確認します。
   - httpsd.conf
   - reverse_proxy.conf
   - proxy_balance.conf

   これらのファイルは次のディレクトリーに格納されています。

   Application Serverのインストールディレクトリー/javaee/glassfish/nodes/*ノード名*/*Webサーバ名*/root/conf/

4.9.4 Administration Console を利用してシステム設定情報を確認する

Administration Console を利用してシステム設定情報を確認するには、[構成管理] タブを使用します。画面の内容が、設定したとおりになっているかを確認します。

前提条件

- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

操作手順

1. Web ブラウザーを起動し、次に示す URL に接続して Administration Console を起動します。

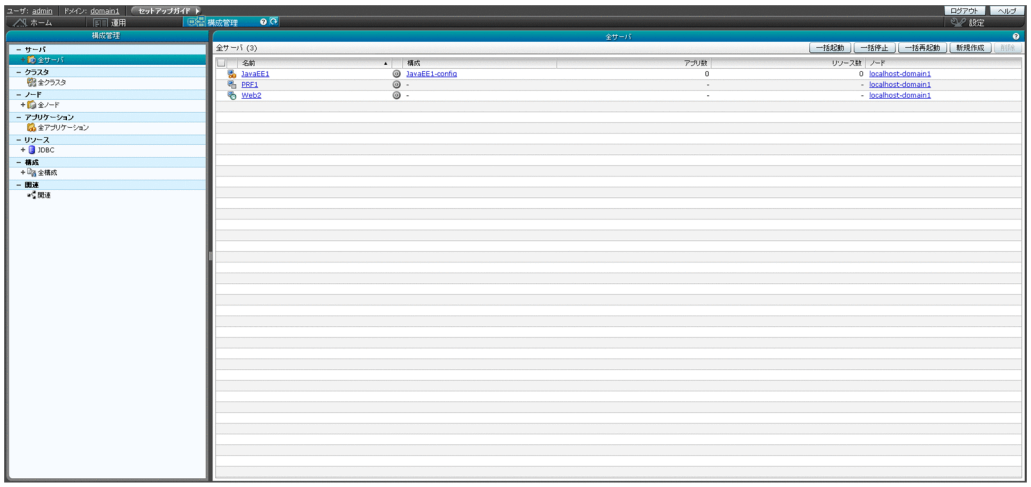
```
http://ドメイン管理サーバのIPアドレス:ドメイン管理サーバのHTTPポート番号/admin/
```

> **メモ**
>
> *ドメイン管理サーバの IP アドレス*のデフォルト値は127.0.0.1、*ドメイン管理サーバの HTTP ポート番号*のデフォルト値は8080 です。どちらもデフォルトの場合は、スタートメニュー/スタート画面から Administration Console を起動できます。
>
> ただし、Windows Server 2012、Windows Server 2012 R2、または Windows 8 でビルトイン Administrator アカウントを使用している場合は、セキュリティ上の理由で Modern UI 版 Internet Explorer を起動できないため、別のアカウントでサインインしてやり直すように促すエラーメッセージが表示されることがあります。 その場合は、Internet Explorer の [ツール] メニューから [インターネット オプション] を選択し、[プログラム] タブの [リンクの開き方を選択] で [常にデスクトップ用 Internet Explorer で開く] を選択して、Administration Console を再度起動してください。

2. [ユーザ ID] テキストボックスにユーザ ID、[パスワード] テキストボックスにパスワードを入力して、[ログイン] ボタンをクリックします。

> **メモ**
>
> ユーザ ID のデフォルト値はadmin、パスワードのデフォルト値はありません。

3. Administration Console の [構成管理] タブを開きます。



4. Java EE サーバ、パフォーマンストレーサー、および Web サーバの設定情報を確認する場合は、次の手順で実施します。
   a. [構成管理] タブのナビゲーションペインのツリーから、[全サーバ] をクリックします。
   b. [全サーバ] ペインの [名前] カラムで、サーバの名称のリンクをクリックします。
   c. [基本情報] タブで、サーバのシステム設定情報が設定したとおりになっているかを確認します。
5. クラスターの設定情報を確認する場合は、次の手順で実施します。
   a. [構成管理] タブのナビゲーションペインのツリーから、[全クラスタ] をクリックします。
   b. [全クラスタ] ペインの [名前] カラムで、クラスター名のリンクをクリックします。

c. [基本情報] タブで、クラスターのシステム設定情報が設定したとおりになっているかを確認します。

6. サーバ間関連の設定情報を確認する場合は、次の手順で実施します。

a. [構成管理] タブのナビゲーションペインのツリーから [関連] をクリックします。

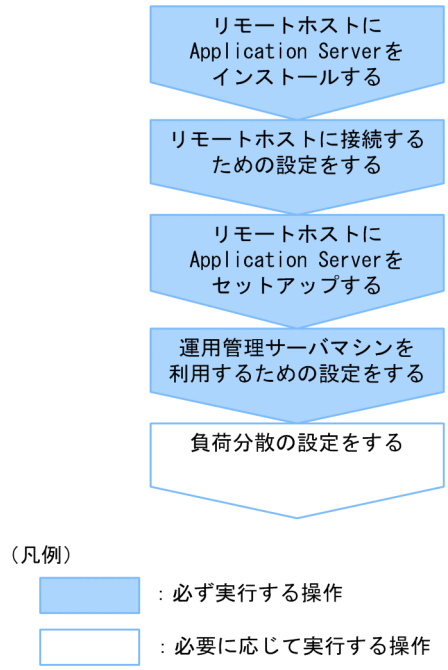
b. [関連] ペインで、サーバ間関連のシステム設定情報が設定したとおりになっているかを確認します。

4.10　リモートホストへの Application Server の構築

ドメイン管理サーバを Application Server と切り分けて、運用管理サーバマシンとして利用するための設定手順、およびクラスター構成のシステムを構築する手順について説明します。クラスター構成とは、複数の Java EE サーバをグループ化して管理するための構成のことです。Application Server をクラスター構成にすると、リクエストが複数の Application Server へ振り分けられるようになり、システムの信頼性や可用性を確保できます。負荷分散には、ソフトウェアロードバランサー、またはハードウェアロードバランサーを利用します。

4.10.1　リモートホストへのアプリケーション実行環境の構築の流れ

運用管理サーバマシンおよびクラスター構成のシステムを構築するために実施する作業と、作業の流れについて説明します。クラスター構成とは、複数の Java EE サーバをグループ化して管理するための構成のことです。運用管理サーバマシンおよびクラスター構成のシステムを構築するには、リモートホストに Application Server をインストール、およびセットアップします。また、ローカルホストを運用管理サーバマシンとして利用するための設定をします。



関連項目

- 4.10.2　リモートホストに Application Server をインストールする
- 4.10.3　リモートホストに接続するための設定をする
- 4.10.4　リモートホストに Application Server をセットアップする
- 4.10.5　運用管理サーバマシンを利用するための設定をする
- 5.1　システム周辺環境の設定について

4.10.2　リモートホストに Application Server をインストールする

新たに用意したリモートホストに Application Server をインストールするには、製品の提供媒体から日立 PP インストーラーを起動し、PP インストール画面でインストールするプログラムを選択します。インストール完了後、環境変数の設定をします。

前提条件

- 管理者特権（スーパーユーザー）で実行する
- リモートホストが起動している
- リモートホストで前提としている OS（OS のパッチを含む）のインストールが完了している

想定ユーザー

- システム構築者

操作手順

1. 次の表に示すパッケージがインストールされていることを確認します。

表 4-5　OS ごとに適用するパッケージ

パッケージ名	Red Hat Enterprise Linux 5	Red Hat Enterprise Linux Server 6	Red Hat Enterprise Linux Server 7
compat-libstdc++-296 (i386)	○	—	—
compat-libstdc++-33 (i386)	○	—	—
compat-libstdc++-33 (i686)	—	○	—
compat-libstdc++-33 (x86_64)	○	○	—
coreutils (x86_64)	○	○	○
findutils (x86_64)	○	○	○
gdb (x86_64)	○	○	○
glibc (i686)	○	○	○※1
glibc (i686) 2.5-24 以降	○※2	—	—
glibc (x86_64)	—	○	○
glibc (x86_64) 2.5-24 以降	○※2	—	—
glibc-common (x86_64)	—	○	○
glibc-common (x86_64) 2.5-24 以降	○※2	—	—
glibc-devel (i386) 2.5-24 以降	○※2	—	—
glibc-devel (i686)	—	○	○※1

パッケージ名	Red Hat Enterprise Linux 5	Red Hat Enterprise Linux Server 6	Red Hat Enterprise Linux Server 7
glibc-devel (x86_64)	—	○	○
glibc-devel (x86_64) 2.5-24 以降	○※2	—	—
glibc-headers (x86_64)	—	○	○
glibc-headers (x86_64) 2.5-24 以降	○※2	—	—
glibc-utils (x86_64)	—	○	○※1
glibc-utils (x86_64) 2.5-24 以降	○※2	—	—
gzip (x86_64)	○	○	○
ksh (x86_64)	—	○	○※1
libgcc (i386)	○	—	—
libgcc (i686)	—	○	○※1
libstdc++ (i386)	○	—	—
libstdc++ (i686)	—	○	○※1
lksctp-tools (x86_64)	○	○	○※1
ncompress (x86_64)	○	○※3	○※1
ncurses (x86_64)	○	○	○
net-tools (x86_64)	○	○	○
nscd (x86_64)	—	○	○※1
nscd (x86_64) 2.5-24 以降	○※2	—	—
nss-softokn-freebl (i686)	—	○	○※1
procps (x86_64)	○	○	○
rpm (x86_64)	○	○	○
sysstat (x86_64)	○	○	○
tar (x86_64)	○	○	○
tcsh (x86_64)	○	○	○※1

(凡例)

○：適用します。

—：該当しません。

注

AIX のパッケージについては、リリースノートを参照してください。

注※1

これらのパッケージについては、使用している Red Hat Enterprise Linux Server 7 のバージョンによってはデフォルトでインストールされない場合があります。

注※2

これらのパッケージについては、使用している Red Hat Enterprise Linux 5 のバージョンによってはインストールメディアに含まれていない場合があります。

注※3

Red Hat Enterprise Linux Server 6 のバージョンが 6.2 の場合、ncompress (x86_64)のバージョンは 4.2.4-54.el6_2.1 以降です。

パッケージがインストールされているかどうかは、rpm コマンドを実行して確認します。コマンドの実行例と実行結果の例を次に示します。

(実行例)

```
#rpm -q --qf  '%{NAME}-%{ARCH}¥n'  ncompress
```

(実行結果の例)

```
ncompress-x86_64
```

この例では、64 ビット版の ncompress パッケージがインストールされていることを示します。

「package *パッケージ名* is not installed」メッセージが表示された場合は、パッケージがインストールされていないので、*パッケージ名*に示すパッケージをインストールしてください。なお、インストール時には、関連するパッケージも必要に応じてインストールしてください。

2. 日立 PP インストーラー実行時の言語種別と、実行するターミナルの言語が一致しているかどうかを確認し、一致していない場合は一致させます。

3. 製品の提供媒体を該当するドライブにセットします。

4. mount コマンドを実行して、該当する媒体のファイルシステムをマウントします。

AIX の場合

```
mount -r -v cdrfs デバイススペシャルファイル名
該当する媒体のファイルシステムのマウントディレクトリー名
```

Linux の場合

```
mount -r -o mode=0544 /dev/cdrom
該当する媒体のファイルシステムのマウントディレクトリー名
```

デバイススペシャルファイル名、および*該当する媒体のファイルシステムのマウントディレクトリー名*は、OS、ハードウェア、および環境によって異なります。

5. 該当する媒体のセットアッププログラムを起動します。

AIX の場合

```
該当する媒体のファイルシステムのマウントディレクトリー名/AIX/setup -m
該当する媒体のファイルシステムのマウントディレクトリー名
```

Linux の場合

```
該当する媒体のファイルシステムのマウントディレクトリー名/x64lin/setup -m
該当する媒体のファイルシステムのマウントディレクトリー名
```

重要

該当する媒体のディレクトリー名やファイル名は、マシン環境によって記述した内容と見え方が異なることがあります。ls コマンドで確認し、表示されたファイル名をそのまま入力してください。

該当する媒体のセットアッププログラムによって、日立 PP インストーラーと常駐プロセス自動起動プログラムがハードディスク上にインストールされ、日立 PP インストーラーが自動的に起動されます。

6. 日立 PP インストーラーのメインメニューで、[I] キーを押します。

7. PP インストール画面で、インストールする製品（Hitachi Application Server）にカーソルを移動させ、[スペース] キーを押します。

製品名の左側には、選択状態を示す<@>が表示されます。

8. 製品を構成するプログラムを選択してインストールする場合、インストールしないプログラムにカーソルを移動させ、[スペース] キーを押します。

インストールが任意のプログラムの左側には、[@]が表示されます。[スペース] キーを押すと、[ ]に表示が変わり、インストール対象から外されます。

なお、インストールが必須のプログラムの左側には、@が表示されます。

9. インストールする製品の左側に<@>、およびプログラムの左側に[@]または@が表示されていることを確認して、[I] キーを押します。

画面の最下行に、Install PP? (y: install, n: cancel)==>が表示されます。

10. [y] キーまたは [Y] キーを押します。

画面の最下行に、Enter the installation path ==>*インストール先のデフォルト値*が表示されます。

[Ctrl] ＋ [n] キーまたは [Ctrl] ＋ [N] キーを押すと、インストールが中止されて PP インストール画面に戻ります。

11. *インストール先のデフォルト値*を確認し、必要に応じてインストール先を変更して、[Enter] キーを押します。

インストール先には半角英数字、-、_以外は指定できません。

インストール先ディレクトリーが存在しない場合、ディレクトリー作成を確認するメッセージが表示されます。

```
The specified path does not exist. Do you want to create the path?
Specified path: インストール先
```

*インストール先*を確認して、[y] キーまたは [Y] キーを押してください。

画面の最下行に、Enter a display name ==>*表示名のデフォルト値*が表示されます。

インストール先を再設定する場合は、[Ctrl] ＋ [B] キーを押します。

12. *表示名のデフォルト値*を確認し、必要に応じて表示名を変更して、[Enter] キーを押します。

表示名に指定できる文字の長さは、半角で 40 文字です。ただし、半角英数字、-、_以外は指定できません。

画面に、指定した表示名、およびインストール先でインストールを続行してよいかどうかを確認するメッセージが表示されます。

```
This program product will be installed at the specified path,
with the specified display name.
Do you want to continue?
Specified display name: 表示名
Specified path        : インストール先
```

表示名、およびインストール先を再設定する場合は、[n] キーまたは [N] キーを押すと、手順 10 に戻ります。

13. 表示名、およびインストール先を確認して、[y] キーまたは [Y] キーを押します。

インストールが開始されます。

14. インストール終了を示すメッセージが出力されたら、[Q] キーを押します。

15. 日立 PP インストーラーのメインメニューで、[L] キーを押します。

PP 一覧表示画面にインストール済みの製品一覧が表示されます。

16. 製品がインストールされていることを確認して、[Q] キーを押します。

17. 日立 PP インストーラーのメインメニューで、[Q] キーを押します。

日立 PP インストーラーが終了し、Application Server のインストールが完了します。

18. 設定ファイル（asenv.conf）に TZ 環境変数を設定します。

設定ファイル（asenv.conf）は、*Java EE サーバのインストールディレクトリー*/glassfish/config/asenv.conf に格納されています。設定ファイル（asenv.conf）の定義例を次に示します。

```
TZ=JST-9
```

19. /etc/hosts ファイルに、自ホスト名に対してネットワークインターフェースに割り当てられた適切な IP アドレスを設定して、自ホスト名から IP アドレスを解決できるように設定します。

20. リクエストの振り分け先として必要なリモートホストごとに、手順 1 から手順 19 までを繰り返します。

次の作業

- リモートホストに接続するための設定をする

関連項目

- 4.10.3　リモートホストに接続するための設定をする

4.10.3　リモートホストに接続するための設定をする

クラスター構成にするためには、ローカルホストとリモートホストが SSH で接続できるようにします。ローカルホストでは、asadmin ユーティリティーコマンドのsetup-ssh サブコマンドで SSH 接続をするための準備をします。

前提条件

- ドメイン管理サーバが起動している
- ローカルホストで Application Server のセットアップが完了している
- リモートホストが起動している
- リモートホストで Application Server のインストールが完了している
- リモートホストとローカルホストが SSH で接続できる

想定ユーザー

- システム構築者

操作手順

1. リモートホストで、ドメイン管理サーバ（ローカルホスト）のホスト名を名前解決できるように hosts ファイルを編集します。
2. ローカルホストで、パスワードファイルを作成します。

```
AS_ADMIN_SSHPASSWORD=リモートホストのパスワード
```

3. ローカルホストで、asadmin ユーティリティーコマンドのsetup-ssh サブコマンドを実行して、暗号キー（SSL キー）を作成します。

```
asadmin --passwordfile パスワードファイルのパス setup-ssh --sshuser リモートホストのSSH ユーザー名 リモートホスト名
```

コマンドの実行結果を次に示します。

```
Command setup-ssh executed successfully.
```

4. ローカルホストで、asadmin ユーティリティーコマンドのcreate-password-alias サブコマンドを実行して、パスワードに対するエイリアスを設定します。

```
asadmin create-password-alias パスワードエイリアス名
```

パスワードが求められた場合は、リモートホストのパスワードを入力してください。

コマンドの実行結果を次に示します。

```
Command create-password-alias executed successfully.
```

5. ドメイン管理サーバを再起動します。

```
asadmin restart-domain
```

コマンドの実行結果を次に示します。

```
Command restart-domain executed successfully.
```

6. ローカルホストで、手順 2 で作成したパスワードファイルを編集します。

```
AS_ADMIN_SSHPASSWORD=${ALIAS=パスワードエイリアス名}
```

7. リクエストの振り分け先として必要なリモートホストごとに、手順 1 から手順 6 までを繰り返します。

次の作業

- リモートホストに Application Server をセットアップする

関連項目

- 4.10.4　リモートホストに Application Server をセットアップする

4.10.4　リモートホストに Application Server をセットアップする

リモートホストに Application Server をセットアップするには、create-node-ssh サブコマンドでリモートホストに対するノードを作成します。そのあと、create-prf サブコマンドでパフォーマンストレーサー、create-instance サブコマンドで Java EE サーバ（サーバインスタンス）をリモートホストに構築します。ハードウェアロードバランサーを利用する場合は、create-webserver サブコマンドでリモートホストに Web サーバを構築します。ソフトウェアロードバランサーを利用する場合は、delete-webserver サブコマンドでローカルホストの Web サーバを削除します。これらのコマンドは、ドメイン管理サーバが起動しているローカルホストで実行します。

前提条件

- ドメイン管理サーバが起動している
- ローカルホストで Application Server のセットアップが完了している
- リモートホストで Application Server のインストールが完了している
- リモートホストとローカルホストが SSH で接続できる

想定ユーザー

- システム構築者

操作手順

1. ローカルホストで、asadmin ユーティリティーコマンドのcreate-node-ssh サブコマンドを実行して、リモートホストに対するノードを追加します。

```
asadmin --user ドメイン管理サーバのユーザー名
--passwordfile パスワードファイルのパス
create-node-ssh --nodehost リモートホストのホスト名
--installdir Application Serverのインストールディレクトリー/javaeeの絶対パス
--sshuser リモートホストのSSHユーザー名
--sshkeyfile ~/.ssh/id_rsa リモートホストのノード名
```

   コマンドの実行結果を次に示します。

```
Command create-node-ssh executed successfully.
```

2. ローカルホストで、asadmin ユーティリティーコマンドのcreate-prf サブコマンドを実行して、リモートホストにパフォーマンストレーサーを構築します。

```
asadmin create-prf --node リモートホストのノード名
リモートホストのパフォーマンストレーサー名
```

   コマンドの実行結果を次に示します。

```
Command create-prf executed successfully.
```

3. ローカルホストで、asadmin ユーティリティーコマンドのcreate-instance サブコマンドを実行して、リモートホストに Java EE サーバ（サーバインスタンス）を構築します。

```
asadmin create-instance --node リモートホストのノード名
--prf リモートホストのパフォーマンストレーサー名
--cluster クラスター名 リモートホストのサーバインスタンス名
```

   *クラスター名*には、ローカルホストで Application Server をセットアップした際に構築したクラスターを指定します。

   コマンドの実行結果を次に示します。

```
Command create-instance executed successfully.
```

4. ハードウェアロードバランサーを利用する場合は、リモートホストに Web サーバを構築します。

   a. ローカルホストで、asadmin ユーティリティーコマンドのcreate-webserver サブコマンドを実行して、リモートホストに Web サーバを構築します。

   ```
   asadmin create-webserver --node リモートホストのノード名
   --prf パフォーマンストレーサー名 リモートホストのWebサーバ名
   ```

   コマンドの実行結果を次に示します。

   ```
   Command create-webserver executed successfully.
   ```

   b. ローカルホストの Web サーバに拡張プロパティーを設定している場合は、ローカルホストで、asadmin ユーティリティーコマンドのset サブコマンドを実行して、リモートホストに作成する Web サーバに拡張プロパティーを設定します。

   > **メモ**
   >
   > 前の手順で実行したcreate-webserver サブコマンドでは、リモートホストに作成する Web サーバに、ローカルホストのサーバテンプレートの設定は適用されますが、拡張プロパティーの設定は適用されません。

   ```
   asadmin set hitachi-webservers.hitachi-webserver.
   リモートホストのWebサーバ名.property.拡張プロパティー名=値
   ```

   コマンドの実行結果を次に示します。

   ```
   hitachi-webservers.hitachi-webserver.リモートホストのWebサーバ名.property.
   拡張プロパティー名=値
   Command set executed successfully.
   ```

   c. ローカルホストで、asadmin ユーティリティーコマンドのcreate-relation サブコマンドを実行して、Web サーバが受信したリクエストのリダイレクト先となる Java EE サーバ（サーバインスタンス）を関連づけます。

   ```
   asadmin create-relation --relationtype redirect
   --from リモートホストのWebサーバ名
   --to リモートホストのサーバインスタンス名
   --properties サーバ間関連のプロパティー名=サーバ間関連のプロパティーの値
   リモートホストのサーバ間関連の関連名
   ```

   リダイレクト関連の関連づけを設定する場合、*サーバ間関連のプロパティー名=サーバ間関連のプロパティーの値*には、path とnetwork-listener を指定します。

   （例）

   ```
   path=/:network-listener=http-listener-1
   ```

コマンドの実行結果を次に示します。

```
Command create-relation executed successfully.
```

d. ローカルホストで静的コンテンツを Web サーバに配置しているときは、リモートホストの Web サーバのドキュメントルートディレクトリーに静的コンテンツを格納します。

リモートホストの Web サーバのドキュメントルートディレクトリーは、*リモートホストの Application Server のインストールディレクトリー*/javaee/glassfish/nodes/*リモートホストのノード名*/*リモートホストの Web サーバ名*/root/htdocs です。

5. リクエストの振り分け先として必要なリモートホストごとに、手順 1 から手順 4 までを繰り返します。

6. Web Server をソフトウェアロードバランサーとして利用する場合は、ローカルホストの Web サーバを削除します。

a. ローカルホストで、asadmin ユーティリティーコマンドのstop-webserver サブコマンドを実行して、ローカルホストの Web サーバを計画停止します。

```
asadmin stop-webserver --graceful true ローカルホストのWebサーバ名
```

コマンドの実行結果を次に示します。

```
Command stop-webserver executed successfully.
```

b. ローカルホストで、asadmin ユーティリティーコマンドのdelete-relation サブコマンドを実行して、不要なリダイレクト関連の関連づけを削除します。

```
asadmin delete-relation ローカルホストのサーバ間関連の関連名
```

リクエストの受信先の Web サーバがソフトウェアロードバランサーとなるため、ローカルホストに構築した Web サーバは不要になります。*ローカルホストのサーバ間関連の関連名* には、ローカルホストの Application Server をセットアップした際に作成したサーバ間関連を指定します。

コマンドの実行結果を次に示します。

```
Command delete-relation executed successfully.
```

c. ローカルホストで、asadmin ユーティリティーコマンドのdelete-webserver サブコマンドを実行して、不要な Web サーバを削除します。

```
asadmin delete-webserver ローカルホストのWebサーバ名
```

ローカルホストの Web サーバ名 には、ローカルホストの Application Server をセットアップした際に構築した Web サーバ名を指定します。

コマンドの実行結果を次に示します。

```
Command delete-webserver executed successfully.
```

次の作業

- 運用管理サーバマシンを利用するための設定をする

関連項目

- 4.10.5　運用管理サーバマシンを利用するための設定をする

4.10.5　運用管理サーバマシンを利用するための設定をする

運用管理サーバマシンは、ドメイン管理サーバだけが動作するマシンです。運用管理サーバマシンを利用するためには、ローカルホストから Application Server の各サーバを削除して、ドメイン管理サーバだけが動作するマシンに変更します。Application Server の各サーバは、asadmin ユーティリティーコマンドのdelete-instance サブコマンド、delete-webserver サブコマンド、およびdelete-prf サブコマンドで削除します。

前提条件

- ドメイン管理サーバが起動している
- リモートホストとローカルホストで Application Server のセットアップが完了している
- リモートホストとローカルホストが SSH で接続できる

想定ユーザー

- システム構築者

操作手順

1. ローカルホストで、asadmin ユーティリティーコマンドのstop-instance サブコマンドを実行して、ローカルホストの Java EE サーバ（サーバインスタンス）を停止します。

```
asadmin stop-instance ローカルホストのサーバインスタンス名
```

コマンドの実行結果を次に示します。

```
Command stop-instance executed successfully.
```

2. ローカルホストで、asadmin ユーティリティーコマンドのstop-prf サブコマンドを実行して、ローカルホストのパフォーマンストレーサーを停止します。

```
asadmin stop-prf ローカルホストのパフォーマンストレーサー名
```

コマンドの実行結果を次に示します。

```
Command stop-prf executed successfully.
```

3. ハードウェアロードバランサーを利用する場合は、ローカルホストで、asadmin ユーティリティーコマンドのdelete-relation サブコマンドを実行して、ローカルホストの Application Server で設定している関連づけを削除します。

```
asadmin delete-relation ローカルホストのサーバ間関連の関連名
```

コマンドの実行結果を次に示します。

```
Command delete-relation executed successfully.
```

4. ローカルホストで、asadmin ユーティリティーコマンドのdelete-instance サブコマンドを実行して、ローカルホストの Java EE サーバ（サーバインスタンス）を削除します。

```
asadmin delete-instance ローカルホストのサーバインスタンス名
```

コマンドの実行結果を次に示します。

```
Command delete-instance executed successfully.
```

5. ハードウェアロードバランサーを利用する場合は、ローカルホストの Web サーバを削除します。

a. ローカルホストで、asadmin ユーティリティーコマンドのstop-webserver サブコマンドを実行して、ローカルホストの Web サーバを計画停止します。

```
asadmin stop-webserver --graceful true ローカルホストのWebサーバ名
```

コマンドの実行結果を次に示します。

```
Command stop-webserver executed successfully.
```

b. ローカルホストで、asadmin ユーティリティーコマンドのdelete-webserver サブコマンドを実行して、ローカルホストの Web サーバを削除します。

```
asadmin delete-webserver ローカルホストのWebサーバ名
```

コマンドの実行結果を次に示します。

```
Command delete-webserver executed successfully.
```

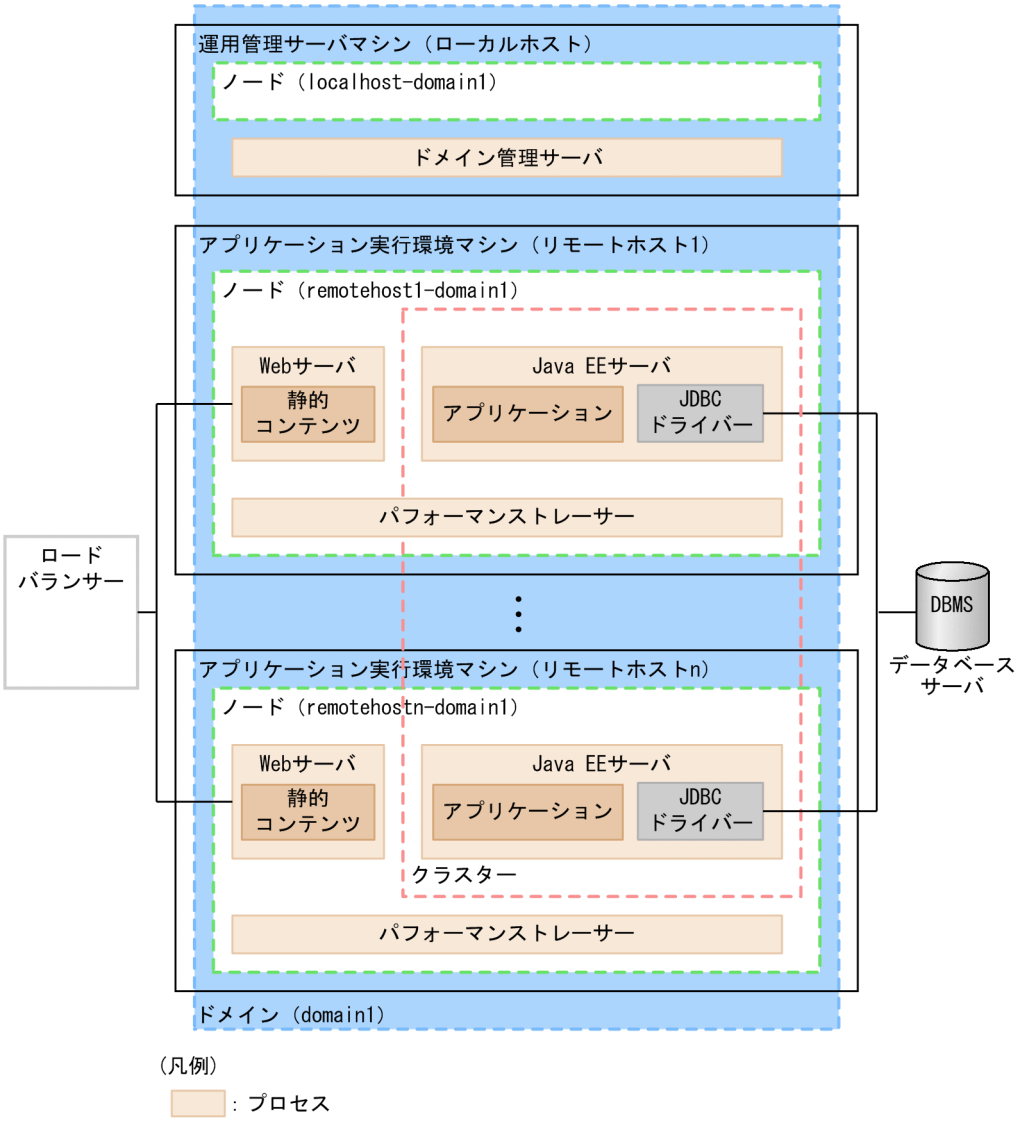
6. ローカルホストで、asadmin ユーティリティーコマンドのdelete-prf サブコマンドを実行して、ローカルホストのパフォーマンストレーサーを削除します。

```
asadmin delete-prf ローカルホストのパフォーマンストレーサー名
```

コマンドの実行結果を次に示します。

```
Command delete-prf executed successfully.
```

構成例を次に示します。


運用管理サーバマシン（ローカルホスト）
ノード（localhost-domain1）
ドメイン管理サーバ
アプリケーション実行環境マシン（リモートホスト1）
ノード（remotehost1-domain1）
Webサーバ
静的
コンテンツ
Java EEサーバ
アプリケーション
JDBC
ドライバー
パフォーマンストレーサー
ロード
バランサー
DBMS
データベース
サーバ
アプリケーション実行環境マシン（リモートホストn）
ノード（remotehostn-domain1）
Webサーバ
静的
コンテンツ
Java EEサーバ
アプリケーション
JDBC
ドライバー
クラスター
パフォーマンストレーサー
ドメイン（domain1）
（凡例）
：プロセス

4.11 Application Server の削除と Application Server のアンインストール

Application Server の削除手順、および Application Server のアンインストール手順について説明します。

4.11.1 Application Server を削除する

Application Server を削除するには、各サーバを停止してから、asadmin ユーティリティーコマンドの delete-relation サブコマンド、delete-instance サブコマンド、delete-webserver サブコマンド、および delete-prf サブコマンドを実行します。各コマンドでは、サーバ間関連、Java EE サーバ、Web サーバ、およびパフォーマンストレーサーを削除します。削除前に、必要に応じて、環境情報のバックアップファイルを取得します。

前提条件

- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドの stop-servers サブコマンドを実行して、パフォーマンストレーサー、サーバインスタンス、および Web サーバを一括停止します。

```
asadmin stop-servers
```

コマンドの実行結果を次に示します。

```
Command stop-servers executed successfully.
```

2. asadmin ユーティリティーコマンドの list-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスが not running になっていることを確認してください。

```
パフォーマンストレーサー名 not running
Command list-prfs executed successfully.
```

3. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがnot running になっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 not running
Command list-instances executed successfully.
```

- *クラスター名*は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。

4. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがnot running になっていることを確認してください。

```
Webサーバ名 not running
Command list-webservers executed successfully.
```

5. 必要に応じて、次の手順で環境情報のバックアップを取得します。

a. asadmin ユーティリティーコマンドのstop-domain サブコマンドを実行して、ドメイン管理サーバを停止します。

```
asadmin stop-domain
```

コマンドの実行結果を次に示します。

```
Command stop-domain executed successfully.
```

b. asadmin ユーティリティーコマンドのbackup-domain サブコマンドを実行して、環境情報のバックアップを取得します。

```
asadmin backup-domain --backupdir バックアップファイルの保存先のディレクトリー
ドメイン名
```

出力されたバックアップファイルを確認してください。

c. asadmin ユーティリティーコマンドのstart-domain サブコマンドを実行して、ドメイン管理サーバを起動します。

```
asadmin start-domain
```

コマンドの実行結果を次に示します。

```
Command start-domain executed successfully.
```

6. ドメインディレクトリー以外で管理しているファイル（サーバテンプレートなど）をバックアップします。

7. asadmin ユーティリティーコマンドのdelete-relation サブコマンドを実行して、サーバ間関連を削除します。

```
asadmin delete-relation サーバ間関連の関連名
```

コマンドの実行結果を次に示します。

```
Command delete-relation executed successfully.
```

8. asadmin ユーティリティーコマンドのdelete-instance サブコマンドを実行して、Java EE サーバ（サーバインスタンス）を削除します。

```
asadmin delete-instance サーバインスタンス名
```

コマンドの実行結果を次に示します。

```
Command delete-instance executed successfully.
```

9. asadmin ユーティリティーコマンドのdelete-webserver サブコマンドを実行して、Web サーバを削除します。

```
asadmin delete-webserver Webサーバ名
```

コマンドの実行結果を次に示します。

```
Command delete-webserver executed successfully.
```

10. asadmin ユーティリティーコマンドのdelete-prf サブコマンドを実行して、パフォーマンストレーサーを削除します。

```
asadmin delete-prf パフォーマンストレーサー名
```

コマンドの実行結果を次に示します。

```
Command delete-prf executed successfully.
```

11. asadmin ユーティリティーコマンドのdelete-node-ssh サブコマンドを実行して、ノードを削除します。

```
asadmin delete-node-ssh ノード名
```

コマンドの実行結果を次に示します。

```
Command delete-node-ssh executed successfully.
```

4.11.2 Application Server をアンインストールする

Application Server をアンインストールするには、日立 PP インストーラーを起動し、PP インストール画面でアンインストールするプログラムを選択します。

前提条件

- 管理者特権（スーパーユーザー）で実行する
- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのstop-servers サブコマンドを実行して、パフォーマンストレーサー、サーバインスタンス、および Web サーバを一括停止します。

```
asadmin stop-servers
```

コマンドの実行結果を次に示します。

```
Command stop-servers executed successfully.
```

2. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがnot running になっていることを確認してください。

```
パフォーマンストレーサー名 not running
Command list-prfs executed successfully.
```

3. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがnot running になっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 not running
Command list-instances executed successfully.
```

- *クラスター名*は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。

4. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがnot running になっていることを確認してください。

```
Webサーバ名 not running
Command list-webservers executed successfully.
```

5. asadmin ユーティリティーコマンドのstop-domain サブコマンドを実行して、ドメインを停止します。

```
asadmin stop-domain ドメイン名
```

コマンドの実行結果を次に示します。

```
Command stop-domain executed successfully.
```

6. 必要に応じて、asadmin ユーティリティーコマンドの backup-domain サブコマンドを実行して、環境情報のバックアップを取得します。

```
asadmin backup-domain --backupdir バックアップファイルの保存先のディレクトリー
ドメイン名
```

出力されたバックアップファイルを確認してください。

7. ドメインディレクトリー以外で管理しているファイル（サーバテンプレートなど）をバックアップします。

8. 次のコマンドを実行して、日立 PP インストーラーを起動します。

AIX の場合

```
/etc/hitachi_setup
```

Linux の場合

```
/etc/hitachi_x64setup
```

9. 日立 PP インストーラーのメインメニューで、[D] キーを押します。

10. 製品（Application Server）を選択してアンインストールする場合は、PP 削除画面で、アンインストールする製品にカーソルを移動させ、[スペース] キーを押します。

製品名の左側には、選択状態を示す<@>が表示され、製品を構成するプログラム名の左側には、選択状態を示す[@]または@が表示されます。

同じ製品が複数インストールされている場合は、アンインストールする製品の表示名（DisplayName）を確認してください。

11. 製品を構成するプログラムを選択してアンインストールする場合は、PP 削除画面で、アンインストールするプログラムにカーソルを移動させ、[スペース] キーを押します。

アンインストールできるプログラムの左側には、[ ]が表示されます。[スペース] キーを押すと、[@] に表示が変わり、アンインストール対象となります。

12. アンインストールする製品の左側に<@>、プログラムの左側に[@]が表示されていることを確認して、[D] キーを押します。

画面の最下行に「Delete PP? (y: delete, n: cancel)==>」メッセージが表示されます。

13. [y] キーまたは [Y] キーを押します。

アンインストールが開始されます。

[n] キーまたは [N] キーを押すと、アンインストールが中止されます。

14. アンインストール終了を示すメッセージが出力されたら、[Q] キーを押します。

15. 日立 PP インストーラーのメインメニューで、[Q] キーを押します。

日立 PP インストーラーが終了し、Application Server のアンインストールが完了します。

5 システム周辺環境の設定

システム周辺環境で必要に応じて設定する項目について説明します。設定する項目は、ネットワーク、負荷分散、および運用の自動化です。

5.1　システム周辺環境の設定について

システム周辺環境で設定が必要な項目は、ネットワーク、負荷分散、および運用の自動化です。ネットワーク設定時の注意事項、負荷分散の種類（ハードウェアロードバランサー、ソフトウェアロードバランサー）ごとの設定内容、および JP1/AJS3 を使用した運用の自動化（ジョブ化）について説明します。

ネットワーク設定時の注意事項

重要

Application Server とデータベースとの間でファイアーウォールの無通信監視機能が有効になっている場合、通信速度が低下するおそれがあるため、無通信監視機能を無効にしてください。

無通信監視機能が有効になっている場合、コネクションプールに確保されているコネクションを使用するときより通信速度が低下することがあります。コネクションプールに確保されているデータベースとのコネクションが無通信監視機能によって切断され、データベースへの操作をするたびに接続が必要となるためです。

負荷分散の設定

- ハードウェアロードバランサーを使用する場合

  ハードウェアロードバランサー上で、Application Server へのリクエスト振り分け、およびセッション維持を設定します。設定の方法については、使用しているハードウェアロードバランサーのマニュアルを参照してください。

  - ドメイン管理サーバと実行環境を別のホストに配置する構成の場合

    構築したリモートホストの Application Server だけにリクエスト振り分けを設定します。ドメイン管理サーバのホストに対するリクエスト振り分けは設定しません。

  - ドメイン管理サーバと実行環境を同じホストに配置する構成の場合

    ドメイン管理サーバのホストに対するリクエスト振り分けを設定します。

- ソフトウェアロードバランサーを使用する場合

  リモートホストのノード上に Web サーバを構築して、ローカルホストと、リモートホストのクラスターとのサーバ間関連を作成することで負荷分散を設定します。

JP1/AJS3 を使用した運用の自動化の設定

Application Server の起動、停止、稼働状況の確認、稼働情報の収集、および障害情報の収集をジョブとして定義し、JP1/AJS3 に登録します。稼働状況の確認、稼働情報の収集、および障害情報の収集については、バッチプログラムの作成も必要です。

関連項目

- 5.2　ソフトウェアロードバランサーを設定する

- 5.3　JP1/AJS3 を使用した運用の自動化を設定する

5.2　ソフトウェアロードバランサーを設定する

リクエストの負荷分散をするために、Web Server をソフトウェアロードバランサーとして使用する場合に必要な設定をします。リモートホストにノードを作成して（create-node-ssh）、ノード上に Web サーバを構築します（create-webserver）。リモートホストの準備が完了したら、ローカルホストと、リモートホストに構築した関連先のクラスターとのサーバ間関連を作成します（create-relation）。

前提条件

- ドメイン管理サーバが起動している
- ソフトウェアロードバランサー用のリモートホストに Application Server がインストールされている
- ソフトウェアロードバランサー用のリモートホストに接続するための設定が完了している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのcreate-node-ssh サブコマンドを実行して、リモートホストにノードを作成します。

```
asadmin --user ドメイン管理ユーザー --passwordfile パスワードファイルのパス
create-node-ssh --nodehost リモートホストのIPアドレス
--sshuser リモートホストのアカウント名 --sshkeyfile ~/.ssh/id_rsa
--installdir Application Serverのインストールディレクトリー/javaeeの絶対パス 作成するノード名
```

コマンドの実行結果を次に示します。

```
Command create-node-ssh executed successfully.
```

2. asadmin ユーティリティーコマンドのcreate-webserver サブコマンドを実行して、リモートホストに作成したノードに Web サーバを構築します。

```
asadmin create-webserver --node 作成したノード名 構築するWebサーバ名
```

コマンドの実行結果を次に示します。

```
Command create-webserver executed successfully.
```

3. asadmin ユーティリティーコマンドのcreate-relation サブコマンドを実行して、サーバ間関連を作成します。

```
asadmin create-relation --relationtype redirect
--from 構築したWebサーバ名
--to 構築済みの関連先のクラスター名
```

```
--properties path=/:network-listener=http-listener-1
作成するサーバ間関連の関連名
```

コマンドの実行結果を次に示します。

```
Command create-relation executed successfully.
```

4. 性能を考慮して静的コンテンツを Web サーバに配置する場合は、Web サーバのドキュメントルートディレクトリーに静的コンテンツを格納します。

   Web サーバのドキュメントルートディレクトリーは、*Application Serverのインストールディレクトリー*/javaee/glassfish/nodes/*ノード名*/*Webサーバ名*/root/htdocs です。

5. 複数のリクエストを送信して、Application Server に正しく振り分けられているかを Web サーバのリクエストログの転送先 IP アドレスとポート番号で確認します。

   次のファイルを参照してください。ファイル名の *x* には可変値が付与されます。

   Application Serverのインストールディレクトリー/javaee/logs/nodes/*ノード名*/*Webサーバ名*/hwsrequest.*x*

   Web サーバのリクエストログの出力例を次に示します。

```
[時刻]（サーバスレッドID）proxy : 転送先IPアドレス:ポート番号 : req send(メッセージテキスト)(メッセージテキスト)
```

5.3　JP1/AJS3 を使用した運用の自動化を設定する

Application Server の起動、停止、稼働状況の確認、稼働情報の収集、および障害情報の収集をジョブとして定義し、JP1/AJS3 に登録します。

前提条件

- JP1/AJS3 の環境が構築されている

想定ユーザー

- システム構築者

操作手順

1. JP1/AJS3 に Application Server の起動をジョブとして定義します。
   a. ドメイン管理サーバ、パフォーマンストレーサー、サーバインスタンス、および Web サーバを起動するコマンドを、ジョブとして定義します。
   b. 作成したジョブを JP1/AJS3 に登録します。
2. JP1/AJS3 に Application Server の停止をジョブとして定義します。
   a. Web サーバ、サーバインスタンス、パフォーマンストレーサー、およびドメイン管理サーバを停止するコマンドを、ジョブとして定義します。
   b. 作成したジョブを JP1/AJS3 に登録します。
3. JP1/AJS3 に Application Server の稼働状況の確認をジョブとして定義します。
   a. 稼働状況を確認するバッチプログラムを作成します。
   b. 作成したバッチプログラムをジョブとして JP1/AJS3 に登録します。
4. JP1/AJS3 に Application Server の稼働情報の収集をジョブとして定義します。
   a. 稼働情報ファイルを収集するバッチプログラムを作成します。
   b. 作成したバッチプログラムをジョブとして JP1/AJS3 に登録します。
5. JP1/AJS3 に Application Server の障害情報の収集をジョブとして定義します。
   a. 障害情報ファイルを収集するバッチプログラムを作成します。
   b. 作成したバッチプログラムをジョブとして JP1/AJS3 に登録します。

関連項目

- 4.6.6　トラブルシュート資料を一括収集するための設定をする
- 7.2.1　コマンドを利用してシステムを開始する

- 7.2.2　コマンドを利用してシステムを停止する
- 7.3.1　コマンドを利用して Application Server の稼働状況を確認する
- 7.3.2　コマンドを利用してデータベースサーバへの接続状況を確認する
- 7.3.3　コマンドを利用してアプリケーションの稼働状況を確認する
- 8.9　システムの利用状況を確認する
- 8.10　システムの動作状況を確認する
- 9.1　Application Server が出力するトラブルシュート資料について

6 高信頼機能の設定

高信頼性を実現するための、障害検知に関する機能（プロセス監視、メッセージ監視）、およびセキュリティーを強化する設定（リバースプロキシー、SSL）について説明します。

6.1　障害検知の設定

システムの障害を検知するために必要な、プロセスの監視とメッセージの監視について説明します。

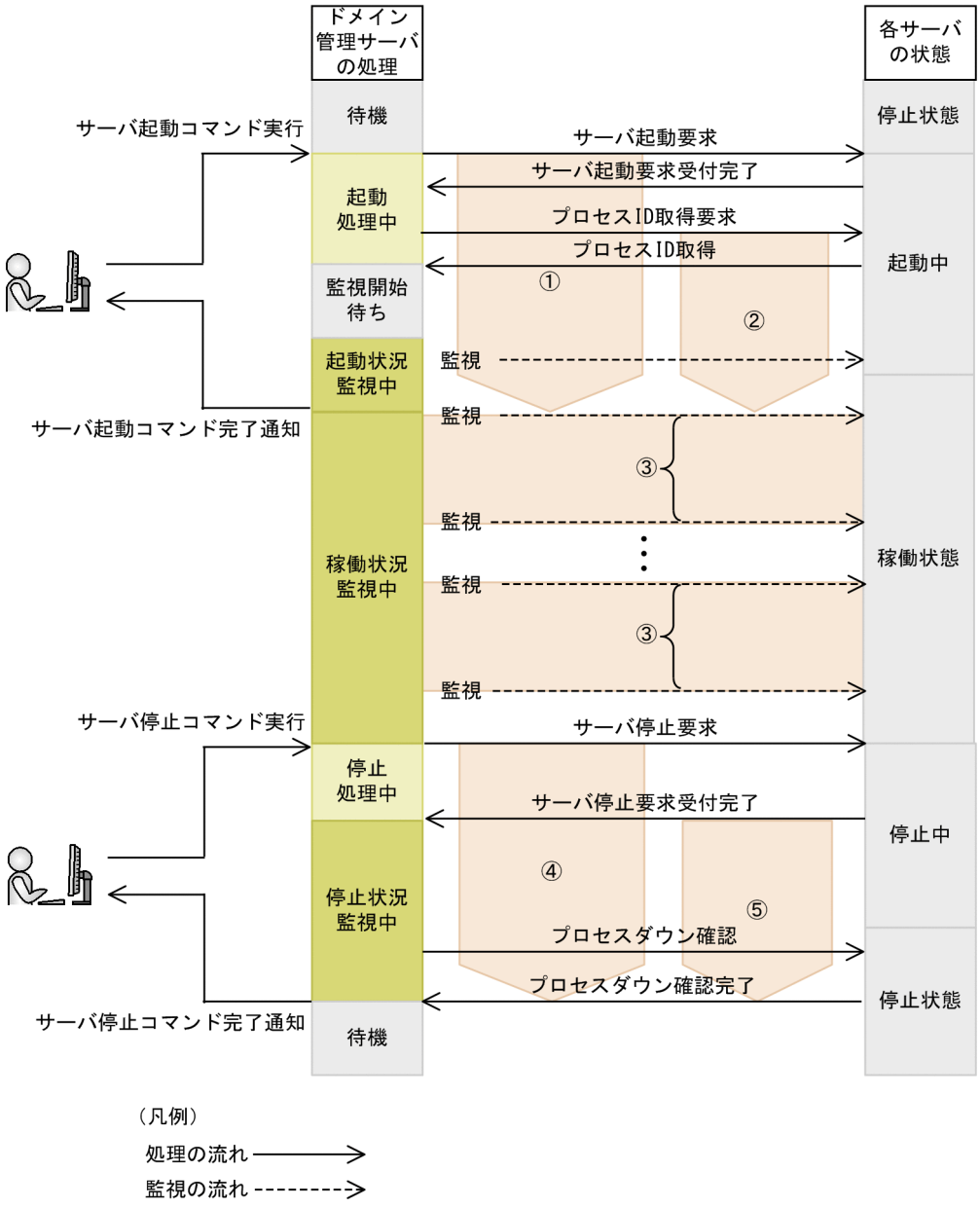
6.1.1　プロセス監視について

ドメイン管理サーバで起動したプロセスは、起動状況、稼働状況、および停止処理が自動的に監視されます。そのほかのプロセスは監視アプリケーションや監視システムで監視できます。

Application Server で障害が発生した場合に短時間で安全に復旧するためには、各プロセスの監視が必要です。ドメイン管理サーバのプロセス監視機能と、監視アプリケーションや監視システムを併用することで、Application Server 全体のプロセスを監視できます。また、ドメイン管理サーバ自体も監視する必要があるため、その方法についても説明します。

ドメイン管理サーバのプロセス監視機能

ドメイン管理サーバが起動している場合は、ドメイン管理サーバのプロセス監視機能によって Java EE サーバ、Web サーバ、パフォーマンストレーサーのプロセスが監視されます。そのため、汎用的なプロセス監視製品や監視システムでプロセスを監視する場合、それぞれのプロセスを監視するのではなく、ドメイン管理サーバだけを監視対象にしてください。ドメイン管理サーバのプロセス監視機能について、次の図にプロセスの流れと、監視のために任意で設定できる時間、および間隔を示します。



図中の設定項目について次の表で説明します。なお、それぞれパラメーターで任意の時間を設定できます。

図中の番号	監視対象のプロセス	設定項目の説明	設定するパラメーター
1	• パフォーマンストレーサー • Web サーバ	起動のタイムアウト時間です。	• パフォーマンストレーサー：hitachi-prf-configs.hitachi-prf-config.*パフォーマンストレーサーの構成名*.hitachi-manage-info.start-timeout-in-seconds • Web サーバ：hitachi-webserver-configs.hitachi-webserver-config.*Webサーバの構成名*.hitachi-manage-
2	JavaEE サーバ		

図中の番号	監視対象のプロセス	設定項目の説明	設定するパラメーター
			info.start-timeout-in-seconds • Java EE サーバ：configs.config.*Java EEサーバの構成名*.hitachi-manage-info.start-timeout-in-seconds
3	• パフォーマンストレーサー • Web サーバ • JavaEE サーバ	各サーバの稼働状況の監視をする間隔です。プロセスダウンやハングアップによるサービス停止などのサーバの障害を監視します。設定した時間に従って、サーバの障害監視の対象期間中、サーバプロセスの稼働状況を定期的に確認します。 稼働状況の確認には監視対象のプロセス 1 つに対して監視用のプロセスを定期的に 1 つ生成します。この処理は CPU リソースを消費するため、オンラインの処理性能に影響を及ぼす場合には、監視する間隔を延ばしてください。	• パフォーマントレーサー：hitachi-prf-configs.hitachi-prf-config.*パフォーマンストレーサーの構成名*.hitachi-manage-info.running-watch-interval-in-seconds • Web サーバ：hitachi-webserver-configs.hitachi-webserver-config.*Webサーバの構成名*.hitachi-manage-info.running-watch-interval-in-seconds • Java EE サーバ：configs.config.*Java EEサーバの構成名*.hitachi-manage-info.running-watch-interval-in-seconds
4	• パフォーマンストレーサー • Web サーバ	停止のタイムアウト時間です。	• パフォーマンストレーサー：hitachi-prf-configs.hitachi-prf-config.*パフォーマンストレーサーの構成名*.hitachi-manage-info.stop-timeout-in-seconds • Web サーバ：hitachi-webserver-configs.hitachi-webserver-config.*Webサーバの構成名*.hitachi-manage-info.stop-timeout-in-seconds • Java EE サーバ：configs.config.*Java EEサーバの構成名*.hitachi-manage-info.stop-timeout-in-seconds
5	JavaEE サーバ		

プロセスの監視可否

監視対象となるプロセスと、プロセスの監視可否を次の表に示します。ドメイン管理サーバで監視できないプロセスは、必要に応じてほかの監視アプリケーションや監視システムで監視できます。

構成要素	監視対象となるプロセス		ドメイン管理サーバでの監視可否	監視アプリケーション、監視システムでの監視可否
アプリケーションクライアント	java		-	○
Web サーバ	httpsd		○※	○
	---	rotatelogs2	-	○
	---	rotatelogs	-	○
	---	httpsd	-	○
JavaEE サーバ	java		○※	○
パフォーマンストレーサー	cprfd		○※	○
ドメイン管理サーバ	java		-	○

（凡例）

○：監視対象にできます。

-：監視対象にできません。

注※

asadmin ユーティリティーコマンドを使用しないで、直接起動した場合はドメイン管理サーバでの監視対象になりません。

ドメイン管理サーバおよび Java EE サーバの監視

ドメイン管理サーバおよび Java EE サーバのプロセス名はjava であるため、ほかの java プロセスと区別できるよう、コマンドのオプションを含めて監視します。コマンドの実行例を次に示します。

（例）ドメイン管理サーバのプロセス監視

```
ps auxww | grep "¥-domainname domain1"
```

（例）Java EE サーバのプロセス監視

```
ps auxww | grep "¥-instancedir .*/localhost-domain1/JavaEE1"
```

6.1.2　メッセージ監視について

Application Server は、障害発生時に特定のメッセージをログファイルに出力します。監視アプリケーションや監視システムを利用して、出力されるメッセージを監視し、メッセージ出力を契機に障害発生を通知するように設定できます。

メッセージは、次の表に示すログファイルに出力されます。監視アプリケーションや監視システムで、ログファイル中の監視対象キーを監視の対象に設定できます。

ログファイルの格納先は、HJES_LOGSDIR 環境変数で変更できます。HJES_LOGSDIR 環境変数のデフォルト値は、"*Application Serverのインストールディレクトリー*/javaee/logs"です。

メッセージが出力されるログファイル			監視対象キー
種別	格納先	ファイル形式	
Web サーバ エラーログ	*HJES_LOGSDIR*/nodes/*ノード名*/*サーバインスタンス名*/error.[*n*]	UPD※1※2	• emerg • alert • crit • error
Java EE サーバ サーバインスタンス メッセージログ	*HJES_LOGSDIR*/nodes/*ノード名*/*サーバインスタンス名*/je_message[*n*].log	SEQ2※3	• KDKDXXXXX-E※4 • EMERGENCY • ALERT • SEVERE • Transaction rolled back due to time out.
ドメイン管理サーバ メッセージログ	*HJES_LOGSDIR*/domains/*ドメイン名*/das_message[*n*].log	SEQ2※3	• KDKDXXXXX-E※4 • EMERGENCY • ALERT • SEVERE

注※1

ファイル名に日付など不定な文字列が設定されるファイル形式です。1 つのログファイルに連続して書き込み続け、ログファイルが一定の容量に達すると、別のファイル名で新たにログファイルを作成して書き込みます。

注※2

ログの分割方法にrotatelogs を使用した場合のファイル形式です。rotatelogs2 を利用した場合のファイル形式は、「WRAP2/新規出現タイプ」です。「WRAP2」は JP1/Base のログファイルトラップ機能のファイル形式です。ラップラウンドするとき、データを削除して再び先頭からログを書き込みます。「新規出現タイプ」は監視対象ファイルの出現タイプです。監視対象ファイルが順に出現します。そのため、監視開始時にすべてのファイルが存在しているわけではありません。

注※3

JP1/Base のログファイルトラップ機能のファイル形式です。ファイル名を変更して保存、またはファイルをいったん削除したあと、同じ名称のファイルを作成して新たにログを書き込みます。

注※4

「XXXXX」の部分はメッセージを出力したプログラムで管理する 5 桁の固有の番号（メッセージ番号）を表します。

関連項目

- 9.1　Application Server が出力するトラブルシュート資料について

6.2　セキュリティー強化の設定

システムのセキュリティーを高めるために、Web サーバに対してリバースプロキシーと SSL の設定をします。なお、この設定方法は、運用管理サーバを経由しないで Web サーバを直接起動する場合に適用する設定方法です。

6.2.1　リバースプロキシーを設定する

セキュリティーの強化のために、httpsd.conf ファイルとreverse_proxy.conf ファイルにリバースプロキシーに関連するパラメーターを設定して、Web サーバをリバースプロキシーとして設置します。

前提条件

- システムのセキュリティー要件が決定している

想定ユーザー

- システム構築者

操作手順

1. リバースプロキシー用のホストに Application Server をインストールします。
2. httpsd.conf ファイルとreverse_proxy.conf ファイルにリバースプロキシーに関連するパラメーターを設定します。
   - httpsd.conf ファイルの定義例

     Include "*Application Serverのインストールディレクトリー*/httpsd/conf/reverse_proxy.conf"
   - reverse_proxy.conf ファイルの定義例

     LoadModule proxy_module modules/mod_proxy.so

     LoadModule proxy_http_module modules/mod_proxy_http.so

     ProxyPass *パス名 バックエンドサーバのURL キー=値*

     ProxyPassReverse *パス名 URL*

     HWSProxyPassReverseCookie *パス名*

6.2.2　SSL を設定する

セキュリティーの強化のために、Web サーバを SSL アクセラレーターとして設置します。httpsd.conf ファイルに SSL アクセラレーターに関連するパラメーターを設定して、Web サーバのコマンド (hwskeygen、hwscertutil reqgen)で、Web サーバの秘密鍵、および証明書発行要求（CSR）を作成します。秘密鍵、および証明書を格納して、設置を完了します。

前提条件

- システムのセキュリティー要件が決定している

想定ユーザー

- システム構築者

操作手順

1. SSL アクセラレーター用のホストに Web サーバをインストールします。
2. httpsd.conf ファイルに SSL アクセラレーターに関連するパラメーターを設定します。
   - httpsd.conf ファイルの定義例

     SSLEnable

     SSLCertificateFile "*Application Serverのインストールディレクトリー*/httpsd/conf/ssl/server/httpsd.pem"

     SSLCertificateKeyFile "*Application Serverのインストールディレクトリー*/httpsd/conf/ssl/server/httpsdkey.pem"
3. hwskeygen コマンドを使用して、Web サーバの秘密鍵を作成します。
   - hwskeygen コマンドの指定例

     hwskeygen -rand *任意のファイル名* -out *秘密鍵ファイル名* -bits *秘密鍵のビット長*
4. hwscertutil reqgen コマンドを使用して、証明書発行要求 （CSR）を作成します。
   - hwscertutil reqgen コマンドの指定例

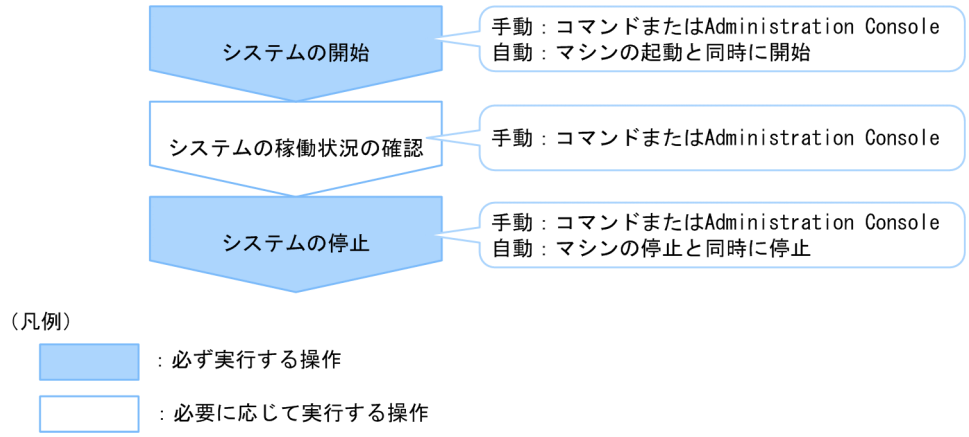
     hwscertutil reqgen -sign *署名アルゴリズム* -key *秘密鍵ファイル名* -out *CSRファイル名*
5. 証明機関に Web サーバの証明書発行を依頼して、証明書を取得します。
6. 秘密鍵、および証明書をhttpsd.conf ファイルで指定した配置先に格納します。

7 通常運用時の作業

運用開始後に日々実施する、通常運用時の作業について説明します。通常運用では、コマンドまたは Administration Console を利用して、システムの開始・停止、およびシステムの稼働状況の確認を実施します。また、マシンの起動・停止と同時に、システムの開始・停止もできます。

7.1　通常運用時の作業の流れ

通常運用では、システムの開始・停止に加えて、システムの稼働状況の確認を実施します。システムの稼働状況の確認では、システムの安定した稼働状態を保つために、Application Server やアプリケーションなどの稼働状況を適宜確認します。システムの開始・停止、およびシステムの稼働状況の確認には、コマンドまたは Administration Console を利用します。コマンドで通常運用時の作業を実行する場合は、管理者権限で実行します。また、マシンの起動・停止と同時に、システムの開始・停止もできます。通常運用時の作業の流れを次に示します。



関連項目

- 7.2.1　コマンドを利用してシステムを開始する
- 7.2.2　コマンドを利用してシステムを停止する
- 7.3.1　コマンドを利用して Application Server の稼働状況を確認する
- 7.3.2　コマンドを利用してデータベースサーバへの接続状況を確認する
- 7.3.3　コマンドを利用してアプリケーションの稼働状況を確認する
- 7.4.1　Administration Console にログインする
- 7.4.2　Administration Console を利用してシステムを開始する
- 7.4.3　Administration Console を利用してシステムを停止する
- 7.5.1　Administration Console を利用して Application Server の稼働状況を確認する
- 7.5.2　Administration Console を利用してデータベースサーバへの接続状況を確認する
- 7.5.3　Administration Console を利用してアプリケーションの稼働状況を確認する
- 7.6.1　マシンの起動時にシステムを同時に開始する（Red Hat Enterprise Linux Server 6 以前および AIX の場合）
- 7.6.2　マシンの起動時にシステムを同時に開始する（Red Hat Enterprise Linux Server 7 の場合）
- 7.6.3　マシンの停止時にシステムを同時に停止する（Red Hat Enterprise Linux Server 6 以前および AIX の場合）

7.2　コマンドを利用したシステムの開始と停止

コマンドを利用したシステムの開始または停止の手順について説明します。

7.2.1　コマンドを利用してシステムを開始する

コマンドを利用してシステムを開始するには、asadmin ユーティリティーコマンドのstart-domain サブコマンドでドメイン管理サーバを起動し、start-servers サブコマンドで Application Server を起動したあと、enable サブコマンドでアプリケーションを開始します。なお、ハードウェアロードバランサーを使用している場合は、最後に、ハードウェアロードバランサーの閉塞を解除します。

前提条件

- ドメイン管理サーバが停止している
- Application Server が停止している
- アプリケーションが停止している

想定ユーザー

- システム運用者

操作手順

1. asadmin ユーティリティーコマンドのstart-domain サブコマンドを実行して、ドメイン管理サーバを起動します。

```
asadmin start-domain
```

コマンドの実行結果を次に示します。

```
Command start-domain executed successfully.
```

2. asadmin ユーティリティーコマンドのstart-servers サブコマンドを実行して、Application Server を一括で起動します。

```
asadmin start-servers
```

コマンドの実行結果を次に示します。

```
Command start-servers executed successfully.
```

3. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがrunningになっていることを確認してください。

```
パフォーマンストレーサー名 running
Command list-prfs executed successfully.
```

4. asadminユーティリティーコマンドのlist-instancesサブコマンドに--longオプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがrunningになっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 running
Command list-instances executed successfully.
```

- *クラスター名*は、複数のJava EEサーバを配置するクラスター構成の場合だけ表示されます。
- サーバインスタンスの起動に成功したかどうかは、Java EEサーバ（サーバインスタンス）のメッセージログにKDKD20031-Iが出力されているかどうか、またはAdministration ConsoleのJava EEサーバ（サーバインスタンス）のステータスでも確認できます。

5. asadminユーティリティーコマンドのlist-webserversサブコマンドを実行して、Webサーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Webサーバのステータスがrunningになっていることを確認してください。

```
Webサーバ名    running
Command list-webservers executed successfully.
```

6. asadminユーティリティーコマンドのenableサブコマンドを実行して、アプリケーションを開始します。

```
asadmin enable --target サーバインスタンス名またはクラスター名 アプリケーション名
```

- 1つのJava EEサーバを配置する構成の場合は、--targetオプションにサーバインスタンス名を指定します。複数のJava EEサーバを配置するクラスター構成の場合は、--targetオプションにクラスター名を指定します。

7. asadminユーティリティーコマンドのlist-applicationsサブコマンドに--longオプションを指定して実行し、アプリケーションの一覧を表示します。

```
asadmin list-applications --long=true サーバインスタンス名またはクラスター名
```

- 1つのJava EEサーバを配置する構成の場合は、サーバインスタンス名を指定します。複数のJava EEサーバを配置するクラスター構成の場合は、クラスター名を指定します。

コマンドの実行結果を次に示します。アプリケーションのステータスがenabled になっていることを確認してください。

```
NAME                    TYPE        STATUS
アプリケーション名  <ear, web>  enabled
アプリケーション名  <web>       enabled
Command list-applications executed successfully.
```

8. Web ブラウザーでアプリケーションの起動 URL を指定して、開始したアプリケーションにアクセスできることを確認します。

```
http://サーバインスタンスのIPアドレス:サーバインスタンスのポート番号/開始したアプリケーションのパス
```

9. ハードウェアロードバランサーを使用している場合は、ハードウェアロードバランサーの閉塞を解除します。

閉塞を解除する方法は、使用しているハードウェアロードバランサーのマニュアルを参照してください。

7.2.2 コマンドを利用してシステムを停止する

コマンドを利用してシステムを停止するには、asadmin ユーティリティーコマンドのdisable サブコマンドでアプリケーションを停止し、stop-servers サブコマンドで Application Server を停止したあと、stop-domain サブコマンドでドメイン管理サーバを停止します。なお、ハードウェアロードバランサーを使用している場合は、最初に、ハードウェアロードバランサーを閉塞します。また、ソフトウェアロードバランサーを使用している場合は、最初に、Web サーバを停止します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している
- アプリケーションが開始している

想定ユーザー

- システム運用者

操作手順

1. ハードウェアロードバランサーを使用している場合は、ハードウェアロードバランサーを閉塞します。

閉塞の方法は、使用しているハードウェアロードバランサーのマニュアルを参照してください。

2. ソフトウェアロードバランサーを使用している場合は、asadmin ユーティリティーコマンドのstop-webserver サブコマンドに--graceful オプションを指定して実行し、Web サーバを計画停止します。

```
asadmin stop-webserver --graceful true Webサーバ名
```

計画停止では、現在受け付けているすべてのリクエストの処理終了を待ってから、Web サーバを停止します。

コマンドの実行結果を次に示します。

```
Command stop-webserver executed successfully.
```

3. ソフトウェアロードバランサーを使用している場合は、asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがnot running になっていることを確認してください。

```
Webサーバ名　 not running
Command list-webservers executed successfully.
```

4. asadmin ユーティリティーコマンドのdisable サブコマンドを実行して、アプリケーションを停止します。

```
asadmin disable --target サーバインスタンス名またはクラスター名 アプリケーション名
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

> **重要**
>
> アプリケーションをアンデプロイするとメモリーからインスタンスが解放されますが、アプリケーションを無効化（disable）した状態ではインスタンスが解放されません。多量のアプリケーションを無効化した場合、使用中のメモリーが増えていて OutOfMemoryError が発生しやすくなります。インスタンスを解放するには、asadmin ユーティリティーコマンドのdelete-application-ref サブコマンドまたはundeploy サブコマンドを実行してください。

コマンドの実行結果を次に示します。

```
Command disable executed successfully.
```

5. asadmin ユーティリティーコマンドのstop-servers サブコマンドを実行して、パフォーマンストレーサー、サーバインスタンス、および Web サーバを一括停止します。

```
asadmin stop-servers
```

コマンドの実行結果を次に示します。

```
Command stop-servers executed successfully.
```

6. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがnot running になっていることを確認してください。

```
パフォーマンストレーサー名 not running
Command list-prfs executed successfully.
```

7. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがnot running になっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 not running
Command list-instances executed successfully.
```

- クラスター名は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。

8. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。なお、ソフトウェアロードバランサーを使用している場合は、この操作は不要です。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがnot running になっていることを確認してください。

```
Webサーバ名 not running
Command list-webservers executed successfully.
```

9. asadmin ユーティリティーコマンドのstop-domain サブコマンドを実行して、ドメイン管理サーバを停止します。

```
asadmin stop-domain
```

コマンドの実行結果を次に示します。

```
Command stop-domain executed successfully.
```

7.3　コマンドを利用したシステムの稼働状況の確認

システムの安定した稼働状態を保つために、Application Server やアプリケーションなどの稼働状況を適宜確認します。コマンドを利用したシステムの稼働状況の確認の手順について説明します。

7.3.1　コマンドを利用して Application Server の稼働状況を確認する

コマンドを利用して Application Server の稼働状況を確認するには、asadmin ユーティリティーコマンドのlist-prfs サブコマンド、list-instances サブコマンド、およびlist-webservers サブコマンドを実行します。

前提条件

- ドメイン管理サーバが起動している

想定ユーザー

- システム運用者

操作手順

1. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

コマンドの実行結果を次に示します。ステータスには、稼働の場合はrunning、停止の場合はnot running が表示されます。

```
パフォーマンストレーサー名　　running
Command list-prfs executed successfully.
```

2. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。ステータスには、稼働の場合はrunning、停止の場合はnot running が表示されます。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 not running
Command list-instances executed successfully.
```

- *クラスター名*は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。

3. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。ステータスには、稼働の場合はrunning、停止の場合はnot running が表示されます。

```
Webサーバ名    running
Command list-webservers executed successfully.
```

7.3.2 コマンドを利用してデータベースサーバへの接続状況を確認する

コマンドを利用してサーバインスタンスからデータベースサーバへの接続状況を確認するには、asadmin ユーティリティーコマンドのping-connection-pool サブコマンドを実行します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している
- サーバインスタンスからデータベースサーバへの接続が設定されている

想定ユーザー

- システム運用者

操作手順

1. asadmin ユーティリティーコマンドのping-connection-pool サブコマンドを実行して、サーバインスタンスからデータベースサーバに接続できるかどうかを確認します。

```
asadmin ping-connection-pool --target サーバインスタンス名 コネクションプールID
```

コマンドの実行結果を次に示します。データベースサーバに接続できる場合は、コマンドが正常に終了します。

```
Command ping-connection-pool executed successfully.
```

7.3.3 コマンドを利用してアプリケーションの稼働状況を確認する

コマンドを利用してサーバインスタンス上のアプリケーションの稼働状況を確認するには、asadmin ユーティリティーコマンドのlist-applications サブコマンドを実行します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している

想定ユーザー

- システム運用者

操作手順

1. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがrunning になっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 running
Command list-instances executed successfully.
```

- *クラスター名*は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。

2. asadmin ユーティリティーコマンドのlist-applications サブコマンドに--long オプションを指定して実行し、サーバインスタンス上のアプリケーションの一覧を表示します。

```
asadmin list-applications --long=true サーバインスタンス名またはクラスター名
```

- 1 つの Java EE サーバを配置する構成の場合は、サーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、クラスター名を指定します。

コマンドの実行結果を次に示します。STATUS でアプリケーションの稼働状況を確認してください。アプリケーションが有効の場合はenabled、無効の場合はdisabled が表示されます。

```
NAME                    TYPE        STATUS
アプリケーション名  <ear, web>  enabled
アプリケーション名  <web>       enabled
Command list-applications executed successfully.
```

7.4 Administration Console を利用したシステムの開始と停止

Administration Console へのログイン、Administration Console を利用したシステムの開始または停止の手順について説明します。

7.4.1 Administration Console にログインする

Administration Console にログインするには、Web ブラウザーから Administration Console を起動し、ユーザー ID とパスワードを入力します。

前提条件

- ドメイン管理サーバが起動している

想定ユーザー

- システム運用者

操作手順

1. Web ブラウザーを起動し、次に示す URL に接続して Administration Console を起動します。

```
http://ドメイン管理サーバのIPアドレス:ドメイン管理サーバのHTTPポート番号/admin/
```

> **メモ**
>
> *ドメイン管理サーバの IP アドレス*のデフォルト値は127.0.0.1、*ドメイン管理サーバの HTTP ポート番号*のデフォルト値は8080 です。どちらもデフォルトの場合は、スタートメニュー/スタート画面から Administration Console を起動できます。
>
> ただし、Windows Server 2012、Windows Server 2012 R2、または Windows 8 でビルトイン Administrator アカウントを使用している場合は、セキュリティ上の理由で Modern UI 版 Internet Explorer を起動できないため、別のアカウントでサインインしてやり直すように促すエラーメッセージが表示されることがあります。 その場合は、Internet Explorer の［ツール］メニューから［インターネット オプション］を選択し、［プログラム］タブの［リンクの開き方を選択］で［常にデスクトップ用 Internet Explorer で開く］を選択して、Administration Console を再度起動してください。

2. [ユーザ ID] テキストボックスにユーザ ID、[パスワード] テキストボックスにパスワードを入力して、[ログイン] ボタンをクリックします。

> **メモ**
>
> ユーザ ID のデフォルト値はadmin、パスワードのデフォルト値はありません。

7.4.2 Administration Console を利用してシステムを開始する

Administration Console を利用してシステムを開始するには、[運用] タブの [全サーバ] ペインで Application Server を起動し、[全アプリケーション] ペインでアプリケーションを開始します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が停止している
- アプリケーションが停止している
- Administration Console にログインしている

想定ユーザー

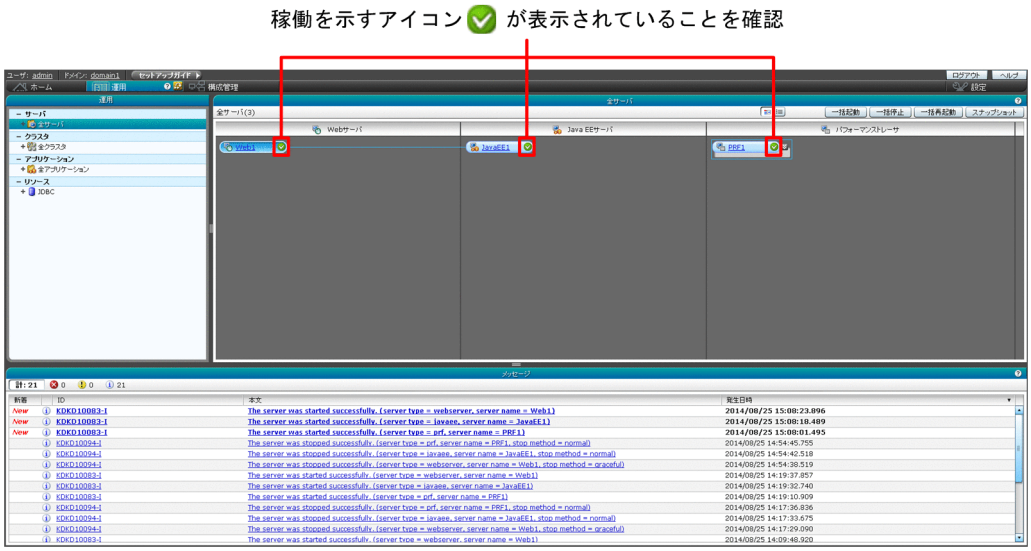
- システム運用者

操作手順

1. Application Server を起動します。
   a. Administration Console の [運用] タブを開き、ナビゲーションペインのツリーから [全サーバ] を選択します。
   b. [全サーバ] ペインの [一括起動] ボタンをクリックします。

c. [サーバの起動] ダイアログボックスで起動するサーバを確認し、[OK] ボタンをクリックします。

[全サーバ] ペインで、すべてのサーバが起動している（稼働を示すアイコン が表示されている）ことを確認してください。



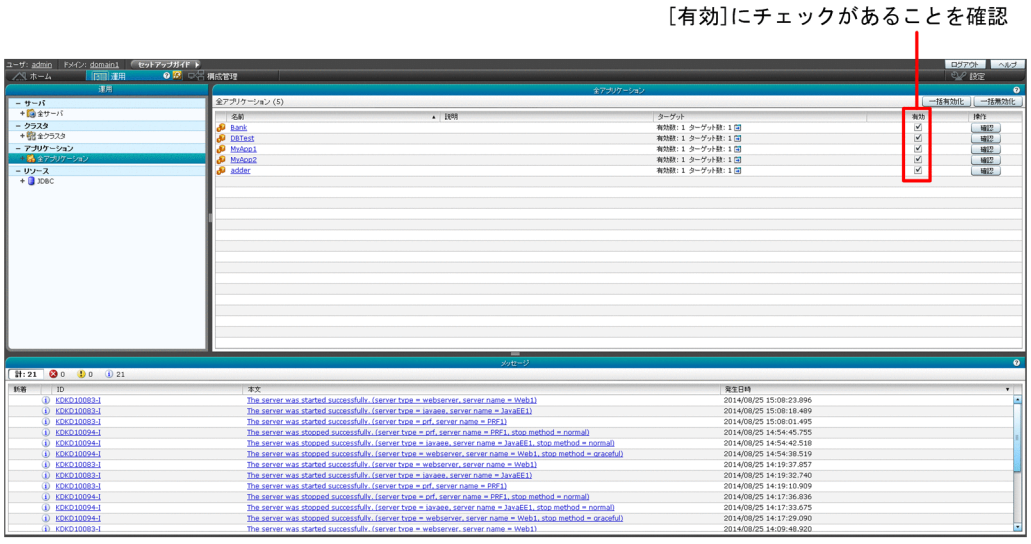
2. アプリケーションを開始します。

a. ナビゲーションペインのツリーから [全アプリケーション] を選択します。

b. [全アプリケーション] ペインの [一括有効化] ボタンをクリックします。

c. [アプリケーションの有効化] ダイアログボックスで開始するアプリケーションを確認し、[OK] ボタンをクリックします。

[全アプリケーション] ペインで、すべてのアプリケーションが開始している（[有効] にチェックがある）ことを確認してください。



3. Web ブラウザーでアプリケーションの起動 URL を指定して、開始したアプリケーションにアクセスできることを確認します。

```
http://サーバインスタンスのIPアドレス:サーバインスタンスのポート番号/開始したアプリケーションのパス
```

7.4.3 Administration Console を利用してシステムを停止する

Administration Console を利用してシステムを停止するには、[運用] タブの［全アプリケーション］ペインでアプリケーションを停止し、[全サーバ] ペインで Application Server を停止します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している
- アプリケーションが開始している
- Administration Console にログインしている

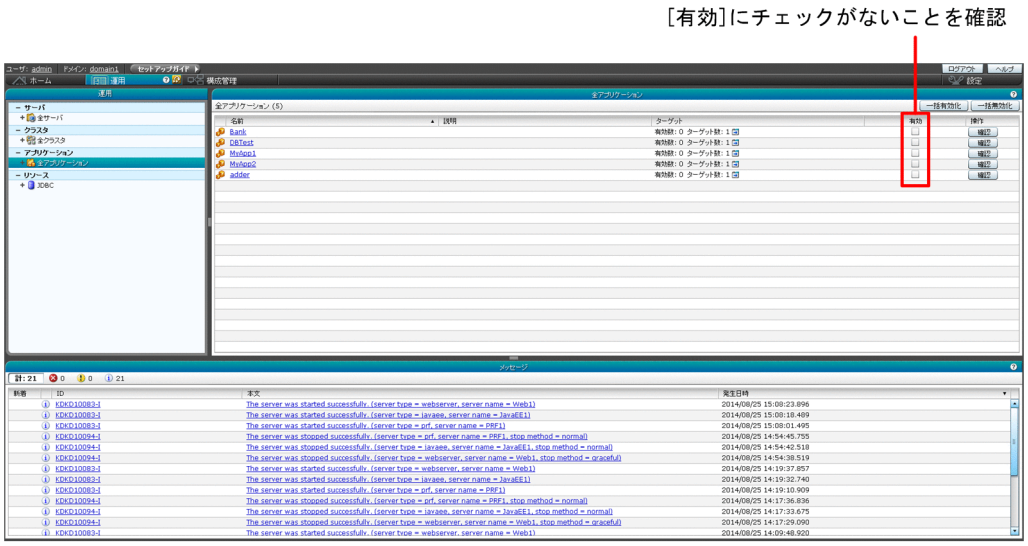
想定ユーザー

- システム運用者

操作手順

1. アプリケーションを停止します。
   a. Administration Console の［運用］タブを開き、ナビゲーションペインのツリーから［全アプリケーション］を選択します。
   b. [全アプリケーション］ペインの［一括無効化］ボタンをクリックします。
   c. [アプリケーションの無効化］ダイアログボックスで停止するアプリケーションを確認し、[OK］ボタンをクリックします。

[全アプリケーション］ペインで、すべてのアプリケーションが停止している（[有効］にチェックがない）ことを確認してください。



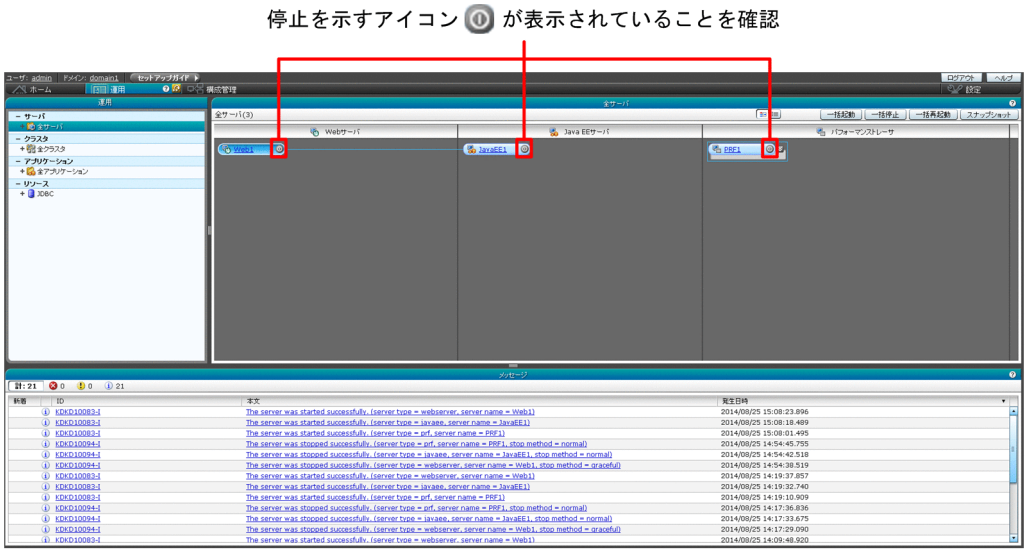
2. Application Server を停止します。

   a. ナビゲーションペインのツリーから［全サーバ］を選択します。

   b. ［全サーバ］ペインの［一括停止］ボタンをクリックします。

   c. ［サーバの停止］ダイアログボックスで停止するサーバを確認し、［OK］ボタンをクリックします。

［全サーバ］ペインで、すべてのサーバが停止している（停止を示すアイコン が表示されている）ことを確認してください。



7.5 Administration Console を利用したシステムの稼働状況の確認

システムの安定した稼働状態を保つために、Application Server やアプリケーションなどの稼働状況を適宜確認します。Administration Console を利用したシステムの稼働状況の確認の手順について説明します。

7.5.1 Administration Console を利用して Application Server の稼働状況を確認する

Administration Console を利用して Application Server の稼働状況を確認するには、［ホーム］タブの［システムのステータス］ペインを使用します。

前提条件

- ドメイン管理サーバが起動している
- Administration Console にログインしている

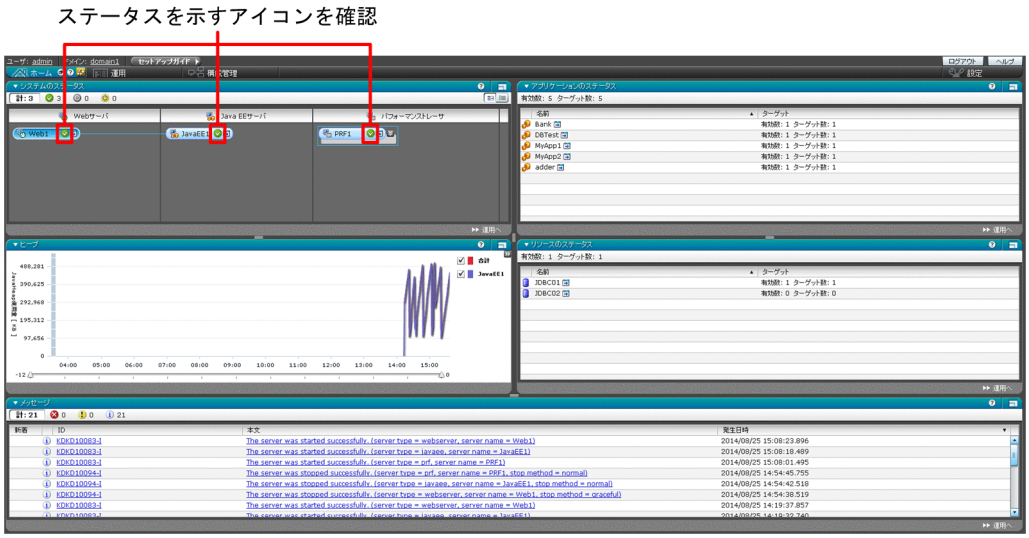
想定ユーザー

- システム運用者

操作手順

1. Administration Console の［ホーム］タブを開きます。

   ［システムのステータス］ペインで、各サーバのステータスを示すアイコン（稼働の場合は  、停止の場合は ）を確認して、Application Server の稼働状況を確認してください。



7.5.2 Administration Console を利用してデータベースサーバへの接続状況を確認する

Administration Console を利用してサーバインスタンスからデータベースサーバへの接続状況を確認するには、［運用］タブの JDBC リソースの［ターゲット ］ペインを使用します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している
- サーバインスタンスからデータベースサーバへの接続が設定されている
- Administration Console にログインしている

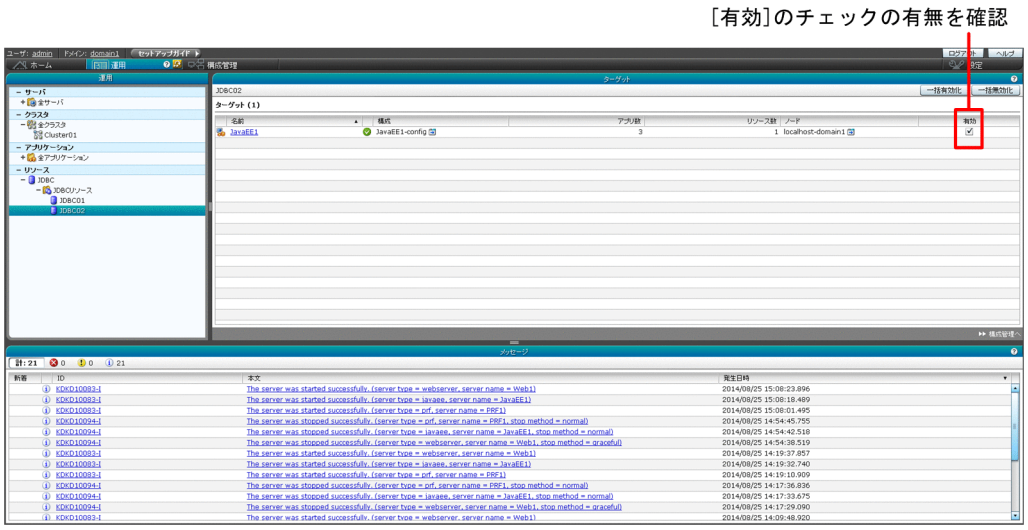
想定ユーザー

- システム運用者

操作手順

1. Administration Console の［運用］タブを開き、ナビゲーションペインのツリーから JDBC リソース名のリンクをクリックします。

   JDBC リソースの［ターゲット］ペインで、JDBC リソースが有効かどうか（[有効］のチェックの有無）を確認してください。



7.5.3 Administration Console を利用してアプリケーションの稼働状況を確認する

Administration Console を利用してアプリケーションの稼働状況を確認するには、［運用］タブのアプリケーションの［ターゲット］ペインを使用します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している
- Administration Console にログインしている

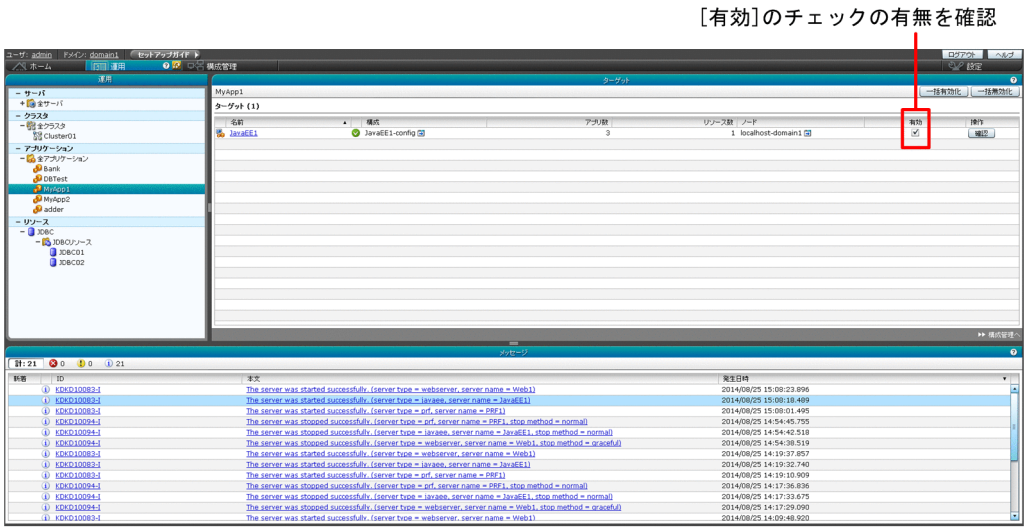
想定ユーザー

- システム運用者

操作手順

1. Administration Console の［運用］タブを開き、ナビゲーションペインのツリーからアプリケーション名のリンクをクリックします。

   アプリケーションの［ターゲット］ペインで、アプリケーションが有効かどうか（［有効］のチェックの有無）を確認してください。



7.6　マシンの起動・停止時のシステムの同時開始・停止

マシンの起動・停止と同時に、システムを開始・停止する手順について説明します。

7.6.1　マシンの起動時にシステムを同時に開始する（Red Hat Enterprise Linux Server 6 以前および AIX の場合）

マシンの起動時にシステムを同時に開始するには、システムの開始処理を記述したスクリプトを作成します。スクリプトには、asadmin ユーティリティーコマンドのstart-domain サブコマンド、およびstart-servers サブコマンドを記述します。

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのstart-domain サブコマンド、およびstart-servers サブコマンドを記述したスクリプトを作成します。
2. 作成したスクリプト（またはそのシンボリックリンク）を rc スクリプトとして UNIX に登録します。
3. マシンの起動時にシステムが同時に開始できるかどうかを確認するために、マシンを再起動します。
4. 実行結果を確認します。

   マシンの起動時にシステムが同時に開始できない場合は、スクリプトの内容や、スクリプトの登録に問題がないかを確認してください。

7.6.2　マシンの起動時にシステムを同時に開始する（Red Hat Enterprise Linux Server 7 の場合）

マシンの起動時にシステムを同時に開始するには、システムの開始処理を記述したsystemd のユニット設定ファイルを作成します。ユニット設定ファイルには、asadmin ユーティリティーコマンドのstart-domain サブコマンド、およびstart-servers サブコマンドを記述します。

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのstart-domain サブコマンド、およびstart-servers サブコマンドを記述したsystemd のユニット設定ファイルを作成します。

   ユニット設定ファイル※1 の記述例を次に示します。

   - start-domain サブコマンドを記述する場合

   ```
   [Unit]
   Description=Application Server Domain
   After=network.target
   [Service]
   Type=oneshot
   ExecStart=/opt/hitachi/APServer/javaee/glassfish/bin/asadmin start-domain※2
   ExecStop=/opt/hitachi/APServer/javaee/glassfish/bin/asadmin stop-domain※2
   RemainAfterExit=yes
   TimeoutStartSec=630
   TimeoutStopSec=90
   [Install]
   WantedBy=multi-user.target
   ```

   - start-servers サブコマンドを記述する場合

   ```
   [Unit]
   Description=Application Server
   After=HAS_domain1.service
   Requires=HAS_domain1.service
   [Service]
   Type=oneshot
   ExecStart=/opt/hitachi/APServer/javaee/glassfish/bin/asadmin start-servers※3
   ExecStop=/opt/hitachi/APServer/javaee/glassfish/bin/asadmin stop-servers※3
   RemainAfterExit=yes
   TimeoutStartSec=930
   TimeoutStopSec=930
   [Install]
   WantedBy=multi-user.target
   ```

   注※1

   コマンドの実行だけをサービスとして記述する場合の例です。systemd のユニット設定ファイルの詳細は、Red Hat Enterprise Linux Server 7 のドキュメントを参照してください。

   注※2

   ローカルホストのデフォルトドメインを起動する場合の例です。asadmin オプションおよびサブコマンドオプションは環境に応じて適切に記述してください。

   注※3

   ローカルホストのデフォルトドメイン内のすべてのサーバを一括起動する場合の例です。asadmin オプションは環境に応じて適切に記述してください。

2. 作成したユニット設定ファイルを/etc/systemd/system/に配置し、systemctl enable *サービス名*コマンドでサービスを有効にします。

3. マシンの起動時にシステムが同時に開始できるかどうかを確認するために、マシンを再起動します。

4. 実行結果を確認します。

   マシンの起動時にシステムが同時に開始できない場合は、スクリプトの内容や、スクリプトの登録に問題がないかを確認してください。

7.6.3 マシンの停止時にシステムを同時に停止する（Red Hat Enterprise Linux Server 6 以前および AIX の場合）

マシンの停止時にシステムを同時に停止するには、システムの停止処理を記述したスクリプトを作成します。スクリプトには、asadmin ユーティリティーコマンドのstop-servers サブコマンド、およびstop-domain サブコマンドを記述します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのstop-servers サブコマンド、およびstop-domain サブコマンドを記述したスクリプトを作成します。

2. 作成したスクリプト（またはそのシンボリックリンク）を rc スクリプトとして UNIX に登録します。

3. マシンの停止時にシステムが同時に停止できるかどうかを確認するために、マシンをシャットダウンします。

4. 実行結果を確認します。

   マシンの停止時にシステムが同時に停止できない場合は、スクリプトの内容や、スクリプトの登録に問題がないかを確認してください。

7.6.4 マシンの停止時にシステムを同時に停止する（Red Hat Enterprise Linux Server 7 の場合）

マシンの停止時にシステムを同時に停止するには、システムの停止処理を記述したsystemd のユニット設定ファイルを作成します。ユニット設定ファイルには、asadmin ユーティリティーコマンドのstop-servers サブコマンド、およびstop-domain サブコマンドを記述します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのstop-servers サブコマンド、およびstop-domain サブコマンドを記述したsystemd のユニット設定ファイルを作成します。

   マシンの起動時にシステムを同時に開始する設定をしていて、かつ記述例に示すとおりstop-servers サブコマンド、stop-domain サブコマンドも記載している場合、手順 1 および手順 2 は不要です。次の記述例は、マシンの起動時にシステムを同時に開始する場合に使用する設定ファイルの例と同じです。

   ユニット設定ファイル※1 の記述例を次に示します。

   - stop-domain サブコマンドを記述する場合

   ```
   [Unit]
   Description=Application Server Domain
   After=network.target
   [Service]
   Type=oneshot
   ExecStart=/opt/hitachi/APServer/javaee/glassfish/bin/asadmin start-domain※2
   ExecStop=/opt/hitachi/APServer/javaee/glassfish/bin/asadmin stop-domain※2
   RemainAfterExit=yes
   TimeoutStartSec=630
   TimeoutStopSec=90
   [Install]
   WantedBy=multi-user.target
   ```

   - stop-servers サブコマンドを記述する場合

   ```
   [Unit]
   Description=Application Server
   After=HAS_domain1.service
   Requires=HAS_domain1.service
   [Service]
   Type=oneshot
   ExecStart=/opt/hitachi/APServer/javaee/glassfish/bin/asadmin start-servers※3
   ExecStop=/opt/hitachi/APServer/javaee/glassfish/bin/asadmin stop-servers※3
   RemainAfterExit=yes
   TimeoutStartSec=930
   TimeoutStopSec=930
   [Install]
   WantedBy=multi-user.target
   ```

注※1

コマンドの実行だけをサービスとして記述する場合の例です。systemd のユニット設定ファイルの詳細は、Red Hat Enterprise Linux Server 7 のドキュメントを参照してください。

注※2

ローカルホストのデフォルトドメインを停止する場合の例です。asadmin オプションおよびサブコマンドオプションは環境に応じて適切に記述してください。

注※3

ローカルホストのデフォルトドメイン内のすべてのサーバを一括停止する場合の例です。asadmin オプションは環境に応じて適切に記述してください。

2. 作成したユニット設定ファイルを/etc/systemd/system/に配置し、systemctl enable *サービス名*コマンドでサービスを有効にします。

3. マシンの停止時にシステムが同時に停止できるかどうかを確認するために、マシンをシャットダウンします。

4. 実行結果を確認します。

マシンの停止時にシステムが同時に停止できない場合は、スクリプトの内容や、スクリプトの登録に問題がないかを確認してください。

8 保守運用時の作業

稼働状況の変化やシステム構成の変更に対応するための保守運用時の作業について説明します。保守運用では、Application Server に対して、稼働状況やシステム構成に応じた環境定義の変更、アプリケーションの入れ替え、環境情報のバックアップ・リストア作業などを実施します。保守運用時の作業には、コマンドまたは Administration Console を利用します。

8.1 保守運用時の作業の概要について

保守運用とは、システムのメンテナンスを目的とした運用です。稼働状況の変化やシステム構成の変更に対応するための作業を実施します。保守運用の作業の目的と作業項目、実行方法の種別など、保守運用時の作業の概要について説明します。

保守運用時の作業一覧

保守運用時の作業の目的と作業項目を次の表に示します。

項番	作業の目的	作業項目
1	次のような場合に、Application Server の環境定義を変更します。 • アクセス数の増加やアプリケーションの変更などで、稼働状況が変わった場合。 • セキュリティーの問題が発生した際の、一時的な対策として実施する場合。 環境定義を変更する場合、一連の作業をする前にパフォーマンストレーサー、サーバインスタンス、および Web サーバを一括停止します。また、一連の作業が完了したら、Application Server を一括起動します。	set サブコマンドを使用して Application Server の設定を変更する
2		サーバテンプレートを使用して Web サーバの設定を変更する
3		create-jvm-options サブコマンドを使用して Java VM オプションを変更する
4		asadmin ユーティリティーコマンドのプロセスに適用する環境変数を変更する
5	アプリケーションへの機能追加や不具合対策などを実施するため、Application Server で稼働しているアプリケーションを入れ替えます。	アプリケーションを入れ替える
6	定期的なメンテナンス時などネットワーク構成を変更する際に、Application Server がインストールされているホストの IP アドレスやホスト名を変更します。	IP アドレスおよびホスト名を変更する
7	Application Server の環境定義の変更に際して設定を保管する場合などに、環境情報をバックアップします。また、環境定義の変更に失敗するなど、バックアップした環境情報を復元する場合にリストアします。	環境情報をバックアップする
8		環境情報をリストアする
9	Application Server の修正パッチが発行された際やシステムの定期メンテナンス時などに、修正パッチおよび修正版を適用します。	修正パッチおよび修正版を適用する
10	マシン構成や環境定義の見直しに向けて、システムの利用状況（アクセスログなど）や動作状況（稼働情報ファイル）を確認します。	システムの利用状況を確認する
11		システムの動作状況を確認する
12	業務処理量の増加が判明した場合や、将来予期される業務処理に備える場合などに、マシンを増設してほかの Application Server と同じ設定をします。	システムをスケールアウトする

項番	作業の目的	作業項目
13	Application Server のバージョンアップ版を適用します。	Application Server をバージョンアップする

保守運用時の作業の実行方法の種別

保守運用時の作業は、次のどちらの方法でも実行できます。このマニュアルでは、コマンドを利用した方法で実行手順を説明しています。

- コマンド
  このマニュアルに記載されている手順のとおりに実行します。コマンドは管理者権限で実行します。
- Administration Console
  Administration Console にログインして、GUI を利用して作業を実行します。

関連項目

- 8.2.1　set サブコマンドを使用して Application Server の設定を変更する
- 8.2.2　サーバテンプレートを使用して Web サーバの設定を変更する
- 8.2.3　create-jvm-options サブコマンドを使用して JavaVM オプションを変更する
- 8.2.4　asadmin ユーティリティーコマンドのプロセスに適用する環境変数を変更する
- 8.3　アプリケーションを入れ替える
- 8.4　IP アドレスおよびホスト名を変更する
- 8.5　環境情報をバックアップする
- 8.6　環境情報をリストアする
- 8.7　修正パッチおよび修正版を適用する
- 8.9　システムの利用状況を確認する
- 8.10　システムの動作状況を確認する
- 8.11　システムをスケールアウトする
- 8.12　Application Server をバージョンアップする

8.2 Application Server の環境定義の変更

Application Server の環境定義として、Application Server の設定、Web サーバの設定、および Java VM オプションの設定を変更できます。

8.2.1 set サブコマンドを使用して Application Server の設定を変更する

Application Server の設定を変更するには、asadmin ユーティリティーコマンドのset サブコマンドで変更します。

前提条件

- ドメイン管理サーバが起動している
- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

操作手順

1. ハードウェアロードバランサーを使用している場合は、ハードウェアロードバランサーを閉塞します。

   閉塞の方法は、使用しているハードウェアロードバランサーのマニュアルを参照してください。

   パフォーマンストレーサー、サーバインスタンス、および Web サーバが停止していない場合は、手順 2 に進んでください。

   パフォーマンストレーサー、サーバインスタンス、および Web サーバがすでに停止している場合は、手順 1～手順 5 を省略してください。

2. asadmin ユーティリティーコマンドのstop-servers サブコマンドを実行して、パフォーマンストレーサー、サーバインスタンス、および Web サーバを一括停止します。

   ```
   asadmin stop-servers
   ```

   コマンドの実行結果を次に示します。

   ```
   Command stop-servers executed successfully.
   ```

3. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

   ```
   asadmin list-prfs
   ```

   コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがnot running になっていることを確認してください。

```
パフォーマンストレーサー名 not running
Command list-prfs executed successfully.
```

4. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがnot running になっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 not running
Command list-instances executed successfully.
```

- クラスター名は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。

5. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがnot running になっていることを確認してください。

```
Webサーバ名 not running
Command list-webservers executed successfully.
```

6. asadmin ユーティリティーコマンドのget サブコマンドを実行して、Application Server の設定値を取得します。

```
asadmin get "*"
```

コマンドの実行結果から、変更対象の設定対象識別子と、変更前の設定値を確認してください。

複数のパラメーターに値が設定されている場合、設定値は次のように扱われます。

パフォーマンストレーサー関連のパラメーターの場合

「hitachi-prfs.hitachi-prf.パフォーマンストレーサー名.で始まるパラメーター」と「hitachi-prf-configs.hitachi-prf-config.パフォーマンストレーサーのコンフィグ名で始まるパラメーター」の両方に値が設定されているときは、「hitachi-prfs.hitachi-prf.パフォーマンストレーサー名.で始まるパラメーター」の値が有効になります。

Web サーバ関連のパラメーターの場合

「hitachi-webservers.hitachi-webserver.Web サーバ名で始まるパラメーター」と「hitachi-webserver-configs.hitachi-webserver-config.Web サーバのコンフィグ名で始まるパラメーター」の両方に値が設定されているときは、「hitachi-webservers.hitachi-webserver.Web サーバ名で始まるパラメーター」の値が有効になります。

サーバインスタンス関連のパラメーターの場合

「servers.server.Java EE サーバ名で始まるパラメーター」と「configs.config.Java EE サーバの構成名で始まるパラメーター」の両方に値が設定されているときは、「servers.server.Java EE サーバ名で始まるパラメーター」の値が有効になります。

7. asadmin ユーティリティーコマンドのset サブコマンドを実行して、変更対象の設定対象識別子に対して変更後の設定値を指定します。

```
asadmin set 設定対象識別子=設定値
```

> **メモ**
>
> Web サーバの標準プロパティー以外（デイレクティブ）の設定値を変更する場合は、サーバテンプレートを使用して Web サーバの設定を変更してください。

コマンドの実行結果を次に示します。

```
Command set executed successfully.
```

8. asadmin ユーティリティーコマンドのget サブコマンドを実行して、Application Server の設定値を再度取得します。

```
asadmin get "*"
```

コマンドの実行結果から、変更対象の設定対象識別子に、手順 7 で指定した設定値が反映されていることを確認してください。

ここで変更作業を完了する場合は、手順 9 に進んでください。

次の環境定義を変更する場合は、Application Server を一括起動しないで、それぞれの環境定義を実施してください。

- Web サーバの設定
- JavaVM オプション
- asadmin ユーティリティーコマンドのプロセスに適用する環境変数

9. asadmin ユーティリティーコマンドのstart-servers サブコマンドを実行して、Application Server を一括で起動します。

```
asadmin start-servers
```

コマンドの実行結果を次に示します。

```
Command start-servers executed successfully.
```

10. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがrunning になっていることを確認してください。

```
パフォーマンストレーサー名 running
Command list-prfs executed successfully.
```

11. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがrunning になっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 running
Command list-instances executed successfully.
```

- クラスター名は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。
- サーバインスタンスの起動に成功したかどうかは、Java EE サーバ（サーバインスタンス）のメッセージログに KDKD20031-I が出力されているかどうか、または Administration Console の Java EE サーバ（サーバインスタンス）のステータスでも確認できます。

12. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがrunning になっていることを確認してください。

```
Webサーバ名    running
Command list-webservers executed successfully.
```

13. ハードウェアロードバランサーを使用している場合は、ハードウェアロードバランサーの閉塞を解除します。

閉塞を解除する方法は、使用しているハードウェアロードバランサーのマニュアルを参照してください。

関連項目

- 8.2.2　サーバテンプレートを使用して Web サーバの設定を変更する
- 8.2.3　create-jvm-options サブコマンドを使用して JavaVM オプションを変更する
- 8.2.4　asadmin ユーティリティーコマンドのプロセスに適用する環境変数を変更する

8.2.2　サーバテンプレートを使用して Web サーバの設定を変更する

Web サーバで標準プロパティー以外の設定値を変更するには、サーバテンプレートを使用します。サーバテンプレートには、Web サーバの運用に必要な設定を記述しています。Web サーバの設定を変更する場合は、拡張プロパティーを設定するか、またはディレクティブをサーバテンプレートに直接記述するかのどちらかを実施します。サーバテンプレートを編集する場合は、拡張プロパティーを設定する方法をお勧めします。

前提条件

- ドメイン管理サーバが起動している
- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

サーバテンプレートの格納先とファイル名

サーバテンプレートのファイル名を次に示します。

- `httpsd.conf@linux.vtl`（Linux の場合）

  Web サーバの基本設定用サーバテンプレートです。リクエスト転送と負荷分散以外の基本設定を記述します。
- `httpsd.conf@aix.vtl`（AIX の場合）

  Web サーバの基本設定用サーバテンプレートです。リクエスト転送と負荷分散以外の基本設定を記述します。
- `reverse_proxy.conf@.vtl`

  Web サーバのリクエスト転送設定用サーバテンプレートです。リクエスト転送先サーバインスタンスがクラスター構成でない場合に、設定を記述します。
- `proxy_balancer.conf@.vtl`

  Web サーバの負荷分散設定用サーバテンプレートです。リクエスト転送先サーバインスタンスがクラスター構成の場合に、設定を記述します。

初回のドメイン起動時、サーバテンプレートの各ファイルは、*Application Serverのインストールディレクトリー*/javaee/glassfish/domains/*ドメイン名*/server_templates/webserver/conf に展開されます。

サーバテンプレートの編集方法

サーバテンプレートは、次の方法で編集できます。

- 拡張プロパティーを設定するために、VTL 構文を記述する

サーバテンプレートにVTL構文を記述し、asadminユーティリティーコマンドのsetサブコマンドの拡張プロパティーの値を操作することで、Webサーバを設定します。

- ディレクティブを直接記述する
  サーバテンプレートにディレクティブを直接記述することで、Webサーバを設定します。

拡張プロパティーを設定するために、VTL構文を記述すると、asadminユーティリティーコマンドのsetサブコマンドでWebサーバの設定を変更し、getサブコマンドで変更内容を確認できるようになります。

サーバテンプレートは、asadminユーティリティーコマンドのcreate-webserverサブコマンド、およびstart-webserverサブコマンド実行時にドメイン管理サーバによって読み込まれ、Webサーバが読み込む定義ファイルに設定が反映されます。

操作手順

1. ハードウェアロードバランサーを使用している場合は、ハードウェアロードバランサーを閉塞します。
   閉塞の方法は、使用しているハードウェアロードバランサーのマニュアルを参照してください。
   パフォーマンストレーサー、サーバインスタンス、およびWebサーバが停止していない場合は、手順2に進んでください。
   パフォーマンストレーサー、サーバインスタンス、およびWebサーバがすでに停止している場合は、手順1～手順5を省略してください。

2. asadminユーティリティーコマンドのstop-serversサブコマンドを実行して、パフォーマンストレーサー、サーバインスタンス、およびWebサーバを一括停止します。

```
asadmin stop-servers
```

   コマンドの実行結果を次に示します。

```
Command stop-servers executed successfully.
```

3. asadminユーティリティーコマンドのlist-prfsサブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

   コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがnot runningになっていることを確認してください。

```
パフォーマンストレーサー名 not running
Command list-prfs executed successfully.
```

4. asadminユーティリティーコマンドのlist-instancesサブコマンドに--longオプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがnot runningになっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 not running
Command list-instances executed successfully.
```

- *クラスター名*は、複数のJava EEサーバを配置するクラスター構成の場合だけ表示されます。

5. asadminユーティリティーコマンドのlist-webserversサブコマンドを実行して、Webサーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Webサーバのステータスがnot runningになっていることを確認してください。

```
Webサーバ名 not running
Command list-webservers executed successfully.
```

6. 拡張プロパティーを設定するために、VTL構文を記述する場合は、次の手順で設定します。

reverse_proxy.confにProxyPreserveHostディレクティブを設定する例で、手順を説明します。

a. テキストエディターなどを利用して、サーバテンプレートのファイル（reverse_proxy.conf@.vtl）を開いて、拡張プロパティーのVTL構文を記述します。

拡張プロパティーは、接頭辞にex_を付加した名称で設定します。

ProxyPreserveHostディレクティブの場合、拡張プロパティーはex_ProxyPreserveHost.valueで設定します。

```
ProxyPreserveHost ${property.ex_ProxyPreserveHost.value}
```

b. サーバテンプレートのファイルを保存します。

c. asadminユーティリティーコマンドのsetサブコマンドを実行して、Webサーバのコンフィグに対して、拡張プロパティー ex_ProxyPreserveHost.value の値にOn を設定します。

```
asadmin set
hitachi-webservers.hitachi-webserver.Webサーバ名.property.ex_ProxyPreserveHost=
On
```

コマンドの実行結果を次に示します。

```
hitachi-webservers.hitachi-webserver.Webサーバ名.property.ex_ProxyPreserveHost=On
Command set executed successfully.
```

d. asadminユーティリティーコマンドのgetサブコマンドを実行して、拡張プロパティー ex_ProxyPreserveHost.value の値が設定したとおりになっているかを確認します。

```
asadmin get
hitachi-webservers.hitachi-webserver.Webサーバ名.property.ex_ProxyPreserveHost
```

コマンドの実行結果を次に示します。

```
hitachi-webservers.hitachi-webserver.Webサーバ名.property.ex_ProxyPreserveHost=On
Command get executed successfully.
```

7. ディレクティブを直接記述する場合は、次の手順で直接記述します。

   a. テキストエディターなどを利用して、サーバテンプレートのファイルを開いて、Web Server のディレクティブを直接記述します。

   ディレクティブをコメントとする場合は、先頭の#の直後に半角スペースを入れて記述します。

   b. サーバテンプレートのファイルを保存します。

   ここで変更作業を完了する場合は、手順 8 に進んでください。

   次の環境定義を変更する場合は、Application Server を一括起動しないで、それぞれの環境定義を実施してください。

   Application Server の設定

   JavaVM オプション

   asadmin ユーティリティーコマンドのプロセスに適用する環境変数

8. asadmin ユーティリティーコマンドのstart-servers サブコマンドを実行して、Application Server を一括で起動します。

```
asadmin start-servers
```

コマンドの実行結果を次に示します。

```
Command start-servers executed successfully.
```

9. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがrunning になっていることを確認してください。

```
パフォーマンストレーサー名 running
Command list-prfs executed successfully.
```

10. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがrunning になっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 running
Command list-instances executed successfully.
```

- クラスター名は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。
- サーバインスタンスの起動に成功したかどうかは、Java EE サーバ（サーバインスタンス）のメッセージログに KDKD20031-I が出力されているかどうか、または Administration Console の Java EE サーバ（サーバインスタンス）のステータスでも確認できます。

11. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがrunning になっていることを確認してください。

```
Webサーバ名    running
Command list-webservers executed successfully.
```

12. ハードウェアロードバランサーを使用している場合は、ハードウェアロードバランサーの閉塞を解除します。

閉塞を解除する方法は、使用しているハードウェアロードバランサーのマニュアルを参照してください。

関連項目

- 8.2.1 set サブコマンドを使用して Application Server の設定を変更する
- 8.2.3 create-jvm-options サブコマンドを使用して JavaVM オプションを変更する
- 8.2.4 asadmin ユーティリティーコマンドのプロセスに適用する環境変数を変更する

8.2.3 create-jvm-options サブコマンドを使用して JavaVM オプションを変更する

JavaVM オプションは、サーバインスタンスと、ドメイン管理サーバに対して指定できます。すでに指定されている JavaVM オプションを変更するには、asadmin ユーティリティーコマンドのdelete-jvm-options サブコマンドで削除してから、create-jvm-options サブコマンドで変更後の JavaVM オプションを指定します。

前提条件

- ドメイン管理サーバが起動している
- Application Server のセットアップが完了している

想定ユーザー

- システム構築者

操作手順

1. ハードウェアロードバランサーを使用している場合は、ハードウェアロードバランサーを閉塞します。

   閉塞の方法は、使用しているハードウェアロードバランサーのマニュアルを参照してください。

   パフォーマンストレーサー、サーバインスタンス、および Web サーバが停止していない場合は、手順 2 に進んでください。

   パフォーマンストレーサー、サーバインスタンス、および Web サーバがすでに停止している場合は、手順 1～手順 5 を省略してください。

2. asadmin ユーティリティーコマンドのstop-servers サブコマンドを実行して、パフォーマンストレーサー、サーバインスタンス、および Web サーバを一括停止します。

   ```
   asadmin stop-servers
   ```

   コマンドの実行結果を次に示します。

   ```
   Command stop-servers executed successfully.
   ```

3. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

   ```
   asadmin list-prfs
   ```

   コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがnot running になっていることを確認してください。

   ```
   パフォーマンストレーサー名 not running
   Command list-prfs executed successfully.
   ```

4. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

   ```
   asadmin list-instances --long=true
   ```

   コマンドの実行結果を次に示します。サーバインスタンスのステータスがnot running になっていることを確認してください。

   ```
   サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 not running
   Command list-instances executed successfully.
   ```

   - *クラスター名*は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。

5. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがnot running になっていることを確認してください。

```
Webサーバ名 not running
Command list-webservers executed successfully.
```

6. asadmin ユーティリティーコマンドのlist-jvm-options サブコマンドを実行して、クラスター内のすべてのサーバインスタンスのオプションの一覧を表示します。

```
asadmin list-jvm-options --target サーバインスタンス名またはクラスター名
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

7. 手順 6 で表示した一覧に変更するオプションがある場合は、asadmin ユーティリティーコマンドのdelete-jvm-options サブコマンドを実行して、変更するオプションを削除します。

```
asadmin delete-jvm-options --target サーバインスタンス名またはクラスター名
[オプション名[=オプションの値][:オプション名[=オプションの値]]...]
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。
- オプションを複数指定する場合は、コロン（:）で区切って指定します。

重要

オプション入力時、記号に対するエスケープ処理が必要になります。

例えば、-XX:MaxMetaspaceSize=192m を指定する場合は、:に対して¥¥を付けてエスケープ処理し、-XX¥¥:MaxMetaspaceSize=192m と入力します。

コマンドの実行結果を次に示します。

```
Deleted n option(s)
Command delete-jvm-options executed successfully.
```

n には、指定したオプションの数に応じて数字が埋め込まれます。

8. asadmin ユーティリティーコマンドのcreate-jvm-options サブコマンドを実行して、すべてのサーバインスタンスに対して、オプションで Java ヒープなどの Java メモリーの値を設定します。

```
asadmin create-jvm-options --target サーバインスタンス名またはクラスター名
[オプション名[=オプションの値][:オプション名[=オプションの値]]...]
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

重要

- すでに指定されているオプションを変更する場合は、delete-jvm-options サブコマンドで変更するオプションを削除してください。削除しないまま、create-jvm-options サブコマンドでオプションを指定した場合は、警告が表示され、同じオプションが複数登録されます。警告が表示された場合は、list-jvm-options サブコマンドで指定されているオプションを確認し、不要なオプションを削除してください。
- ドメイン管理サーバは 1 時間ごと、サーバインスタンスは 24 時間ごとに SystemGC を実行します。SystemGC の実行間隔は、sun.rmi.dgc.server.gcInterval と、sun.rmi.dgc.client.gcInterval で変更できます。これらのシステムプロパティーで FullGC の発生間隔を広げても GC 発生回数が削減されない場合、Java ヒープが不足していることがあります。この場合は、Java ヒープのチューニングを実施することで改善することがあります。

コマンドの実行結果を次に示します。

```
Created n option(s)
Command create-jvm-options executed successfully.
```

n には、指定したオプションの数に応じて数字が埋め込まれます。

9. asadmin ユーティリティーコマンドのlist-jvm-options サブコマンドを実行して、すべてのサーバインスタンスのオプションの一覧を表示します。

```
asadmin list-jvm-options --target サーバインスタンス名またはクラスター名
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

手順 8 で指定したオプションの値が変更されていることを確認してください。

10. asadmin ユーティリティーコマンドのlist-jvm-options サブコマンドを実行して、ドメイン管理サーバのオプションの一覧を表示します。

```
asadmin list-jvm-options
```

11. 手順 10 で表示した一覧に変更するオプションがある場合は、asadmin ユーティリティーコマンドの delete-jvm-options サブコマンドを実行して、変更するオプションを削除します。

```
asadmin delete-jvm-options [オプション名[=オプションの値]
[:オプション名[=オプションの値]]...]
```

コマンドの実行結果を次に示します。

```
Deleted n option(s)
Command delete-jvm-options executed successfully.
```

n には、指定したオプションの数に応じて数字が埋め込まれます。

12. asadmin ユーティリティーコマンドのcreate-jvm-options サブコマンドを実行して、ドメイン管理サーバに対して、オプションで Java ヒープなどの Java メモリーの値を設定します。

```
asadmin create-jvm-options [オプション名[=オプションの値]
[:オプション名[=オプションの値]]...]
```

オプション名[=*オプションの値*]には、オプションとして、-Xms1024m、-Xmx1024m などを設定します。

> **重要**
>
> ドメイン管理サーバの Java ヒープサイズは、デプロイするアプリケーションのアーカイブのサイズに応じて調整してください。アプリケーションのアーカイブのサイズによっては、ドメイン管理サーバの Java ヒープサイズが枯渇し、メモリー不足となることがあります。
>
> また、ドメイン管理サーバの Java ヒープサイズに不適切な値（極端に小さい値や大きい値など）を指定した場合、ドメイン管理サーバが起動しなくなり、ドメインの再構築が必要になることがあります。
>
> このような事態を避けるために、ドメイン管理サーバのオプションを変更する場合は、あらかじめ、backup-domain コマンドを実行して、ドメインのバックアップを取得しておくことをお勧めします。

コマンドの実行結果を次に示します。

```
Created n option(s)
Command create-jvm-options executed successfully.
```

n には、指定したオプションの数に応じて数字が埋め込まれます。

13. asadmin ユーティリティーコマンドのlist-jvm-options サブコマンドを実行して、ドメイン管理サーバのオプションの一覧を表示します。

```
asadmin list-jvm-options
```

手順 12 で指定したオプションの値が変更されていることを確認してください。

14. サーバインスタンス、およびドメイン管理サーバに対して、Java メモリー以外のオプションを指定する場合は、手順 6 から手順 13 までを繰り返します。

ここで変更作業を完了する場合は、手順 15 に進んでください。

次の環境定義を変更する場合は、Application Server を一括起動しないで、それぞれの環境定義を実施してください。

- Application Server の設定
- Web サーバの設定
- asadmin ユーティリティーコマンドのプロセスに適用する環境変数

15. asadmin ユーティリティーコマンドのstart-servers サブコマンドを実行して、Application Server を一括で起動します。

```
asadmin start-servers
```

コマンドの実行結果を次に示します。

```
Command start-servers executed successfully.
```

16. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがrunning になっていることを確認してください。

```
パフォーマンストレーサー名 running
Command list-prfs executed successfully.
```

17. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがrunning になっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 running
Command list-instances executed successfully.
```

- クラスター名は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。
- サーバインスタンスの起動に成功したかどうかは、Java EE サーバ（サーバインスタンス）のメッセージログに KDKD20031-I が出力されているかどうか、または Administration Console の Java EE サーバ（サーバインスタンス）のステータスでも確認できます。

18. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがrunning になっていることを確認してください。

```
Webサーバ名    running
Command list-webservers executed successfully.
```

19. ハードウェアロードバランサーを使用している場合は、ハードウェアロードバランサーの閉塞を解除します。

閉塞を解除する方法は、使用しているハードウェアロードバランサーのマニュアルを参照してください。

関連項目

- 8.2.1　set サブコマンドを使用して Application Server の設定を変更する
- 8.2.2　サーバテンプレートを使用して Web サーバの設定を変更する
- 8.2.4　asadmin ユーティリティーコマンドのプロセスに適用する環境変数を変更する

8.2.4　asadmin ユーティリティーコマンドのプロセスに適用する環境変数を変更する

asadmin ユーティリティーコマンドのプロセスに適用する環境変数を変更するには、Java EE Server の環境変数定義ファイル（asenv.conf）を編集します。asadmin ユーティリティーコマンドのプロセスに適用する環境変数には、Java ヒープなどの Java メモリーの値や、asadmin ユーティリティーコマンドのログなどを設定します。例えば、デプロイしたアプリケーション数やアプリケーション中のファイル数が多く、Java EE サーバの起動時にメモリー不足になる場合は、asadmin ユーティリティーコマンドのプロセスに適用する Java ヒープ領域のサイズを変更します。

前提条件

- ドメイン管理サーバが起動している
- Application Server が起動している

想定ユーザー

- システム構築者

操作手順

1. ハードウェアロードバランサーを使用している場合は、ハードウェアロードバランサーを閉塞します。

   閉塞の方法は、使用しているハードウェアロードバランサーのマニュアルを参照してください。

   パフォーマンストレーサー、サーバインスタンス、および Web サーバが停止していない場合は、手順 2 に進んでください。

   パフォーマンストレーサー、サーバインスタンス、および Web サーバがすでに停止している場合は、手順 1～手順 5 を省略してください。

2. asadmin ユーティリティーコマンドのstop-servers サブコマンドを実行して、パフォーマンストレーサー、サーバインスタンス、および Web サーバを一括停止します。

   ```
   asadmin stop-servers
   ```

   コマンドの実行結果を次に示します。

   ```
   Command stop-servers executed successfully.
   ```

3. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

   ```
   asadmin list-prfs
   ```

   コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがnot running になっていることを確認してください。

   ```
   パフォーマンストレーサー名 not running
   Command list-prfs executed successfully.
   ```

4. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

   ```
   asadmin list-instances --long=true
   ```

   コマンドの実行結果を次に示します。サーバインスタンスのステータスがnot running になっていることを確認してください。

   ```
   サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 not running
   Command list-instances executed successfully.
   ```

   - *クラスター名*は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。

5. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

   ```
   asadmin list-webservers
   ```

コマンドの実行結果を次に示します。Web サーバのステータスがnot running になっていることを確認してください。

```
Webサーバ名 not running
Command list-webservers executed successfully.
```

6. Java EE Server の環境変数定義ファイル（asenv.conf）を編集して、asadmin ユーティリティーコマンドのプロセスに適用する環境変数を変更します。

複数の Java EE サーバを配置するクラスター構成の場合は、リモートホストおよびローカルホストのそれぞれに対して、環境変数を変更します。

Java EE Server の環境変数定義ファイル

Application Serverのインストールディレクトリー/javaee/glassfish/config/asenv.conf

編集の例

例えば、Java ヒープ領域の最大サイズを変更する場合は、環境変数 HJES_ASADMIN_JVM_OPTIONS に値を指定します。

```
HJES_ASADMIN_JVM_OPTIONS=-Xmx256m
```

ここで変更作業を完了する場合は、手順 7 に進んでください。

次の環境定義を変更する場合は、Application Server を一括起動しないで、それぞれの環境定義を実施してください。

- Application Server の設定
- Web サーバの設定
- JavaVM オプション

7. asadmin ユーティリティーコマンドのstart-servers サブコマンドを実行して、Application Server を一括で起動します。

```
asadmin start-servers
```

コマンドの実行結果を次に示します。

```
Command start-servers executed successfully.
```

8. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがrunning になっていることを確認してください。

```
パフォーマンストレーサー名 running
Command list-prfs executed successfully.
```

9. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがrunning になっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 running
Command list-instances executed successfully.
```

- *クラスター名*は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。
- サーバインスタンスの起動に成功したかどうかは、Java EE サーバ（サーバインスタンス）のメッセージログに KDKD20031-I が出力されているかどうか、または Administration Console の Java EE サーバ（サーバインスタンス）のステータスでも確認できます。

10. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがrunning になっていることを確認してください。

```
Webサーバ名    running
Command list-webservers executed successfully.
```

11. ハードウェアロードバランサーを使用している場合は、ハードウェアロードバランサーの閉塞を解除します。

閉塞を解除する方法は、使用しているハードウェアロードバランサーのマニュアルを参照してください。

関連項目

- 8.2.1 set サブコマンドを使用して Application Server の設定を変更する
- 8.2.2 サーバテンプレートを使用して Web サーバの設定を変更する
- 8.2.3 create-jvm-options サブコマンドを使用して JavaVM オプションを変更する

8.3 アプリケーションを入れ替える

アプリケーションへの機能追加や不具合対策などを実施するため、Application Server で稼働するアプリケーションを入れ替えます。この作業には、稼働中の Application Server を停止させたあと、入れ替え対象のアプリケーションをasadmin ユーティリティーコマンドのundeploy サブコマンドでアンデプロイし、変更後のアプリケーションをdeploy サブコマンドでデプロイします。なお、ハードウェアロードバランサーを使用している場合は、最初に、ハードウェアロードバランサーを閉塞します。

前提条件

- ドメイン管理サーバが起動している
- サーバインスタンスが起動している
- サーバインスタンスでアプリケーションが稼働している
- 入れ替えるアプリケーションが存在する

想定ユーザー

- システム構築者

操作手順

1. ハードウェアロードバランサーを使用している場合は、ハードウェアロードバランサーを閉塞します。
   閉塞の方法は、使用しているハードウェアロードバランサーのマニュアルを参照してください。

2. asadmin ユーティリティーコマンドのstop-servers サブコマンドを実行して、パフォーマンストレーサー、サーバインスタンス、および Web サーバを一括停止します。

   ```
   asadmin stop-servers
   ```

   コマンドの実行結果を次に示します。

   ```
   Command stop-servers executed successfully.
   ```

3. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

   ```
   asadmin list-prfs
   ```

   コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがnot running になっていることを確認してください。

   ```
   パフォーマンストレーサー名 not running
   Command list-prfs executed successfully.
   ```

4. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがnot running になっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 not running
Command list-instances executed successfully.
```

- *クラスター名*は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。

5. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがnot running になっていることを確認してください。

```
Webサーバ名 not running
Command list-webservers executed successfully.
```

6. アプリケーション名を確認するためasadmin ユーティリティーコマンドのlist-applications サブコマンドを実行して、サーバインスタンスまたはクラスターを対象に、デプロイされているアプリケーションの一覧を表示します。

```
asadmin list-applications --long=true サーバインスタンス名またはクラスター名
```

- 1 つの Java EE サーバを配置する構成の場合は、サーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、クラスター名を指定します。

コマンドの実行結果を次に示します。出力されたアプリケーションの一覧から、アプリケーション名を確認してください。

```
NAME                    TYPE                STATUS
アプリケーション名    <ear, web, ejb>     enabled
アプリケーション名    <ear, web, ejb>     enabled
Command list-application executed successfully.
```

7. 手順 6 で確認したアプリケーション名を指定してasadmin ユーティリティーコマンドのundeploy サブコマンドを実行し、サーバインスタンスまたはクラスターを対象に、入れ替えるアプリケーションをサーバインスタンスからアンデプロイします。

```
asadmin undeploy --target サーバインスタンス名またはクラスター名 アプリケーション名
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

コマンドの実行結果を次に示します。入れ替えるアプリケーションのアンデプロイに成功したことを確認してください。

```
Command undeploy executed successfully.
```

8. 入れ替えるすべてのアプリケーションに対して、手順 7 の作業を実施します。

9. asadmin ユーティリティーコマンドのdeploy サブコマンドを実行し、サーバインスタンスまたはクラスターを対象に、変更後のアプリケーションをサーバインスタンスへデプロイします。

```
asadmin deploy --target サーバインスタンス名またはクラスター名 アプリケーションのファイルパス
```

- 1 つの Java EE サーバを配置する構成の場合は、--target オプションにサーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、--target オプションにクラスター名を指定します。

コマンドの実行結果を次に示します。変更後のアプリケーションのデプロイに成功したことを確認してください。

```
Application deployed with name アプリケーション名.
Command deploy executed successfully.
```

10. 変更後のすべてのアプリケーションに対して、手順 9 の作業を実施します。

> **メモ**
>
> 手順 6 から手順 10 では、アンデプロイしたあとにデプロイしてアプリケーションを入れ替えていますが、リデプロイしてアプリケーションを入れ替えることもできます。

11. ドメインディレクトリーの下位階層のdocroot ディレクトリーに格納されている静的コンテンツを、新しいコンテンツに入れ替えます。

12. Web サーバのドキュメントルートディレクトリー（*Application Serverのインストールディレクトリー*/javaee/glassfish/nodes/*ノード名*/*Webサーバ名*/root/htdocs）に格納されている静的コンテンツを、新しいコンテンツに入れ替えます。

13. asadmin ユーティリティーコマンドのstart-servers サブコマンドを実行して、Application Server を一括で起動します。

```
asadmin start-servers
```

コマンドの実行結果を次に示します。

```
Command start-servers executed successfully.
```

14. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがrunning になっていることを確認してください。

```
パフォーマンストレーサー名 running
Command list-prfs executed successfully.
```

15. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがrunning になっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 running
Command list-instances executed successfully.
```

- クラスター名は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。
- サーバインスタンスの起動に成功したかどうかは、Java EE サーバ（サーバインスタンス）のメッセージログに KDKD20031-I が出力されているかどうか、または Administration Console の Java EE サーバ（サーバインスタンス）のステータスでも確認できます。

16. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがrunning になっていることを確認してください。

```
Webサーバ名    running
Command list-webservers executed successfully.
```

17. 変更後のアプリケーションの状態を確認するため、asadmin ユーティリティーコマンドのlist-applications サブコマンドを実行して、サーバインスタンスまたはクラスターを対象に、サーバインスタンスにデプロイされているアプリケーションの一覧を表示します。

list-applications サブコマンドを実行する際、アプリケーションが有効かどうかを表示するため--long オプションを指定します。

```
asadmin list-applications --long=true サーバインスタンス名またはクラスター名
```

- 1 つの Java EE サーバを配置する構成の場合は、サーバインスタンス名を指定します。複数の Java EE サーバを配置するクラスター構成の場合は、クラスター名を指定します。

コマンドの実行結果を次に示します。NAME にはアプリケーション名が、TYPE には、アプリケーションの種類が表示されます。すべてのアプリケーションのステータスがenabled（有効）になっていることを確認してください。

```
NAME                      TYPE         STATUS
アプリケーション名   <ear, web>   enabled
アプリケーション名   <web>        enabled
Command list-applications executed successfully.
```

18. ハードウェアロードバランサーを使用している場合は、外部からアプリケーションにアクセスできるようにするため、ハードウェアロードバランサーの閉塞を解除します。

閉塞を解除する方法は、使用しているハードウェアロードバランサーのマニュアルを参照してください。

8.4 IP アドレスおよびホスト名を変更する

定期メンテナンス時などネットワークの構成を変更する場合、Application Server がインストールされているホストの IP アドレスやホスト名を変更します。この作業には、稼働中の Application Server を停止し、ドメイン管理サーバのホストを対象にする場合はドメイン管理サーバを停止させたあと、OS の手順に従って対象のホストの IP アドレスまたはホスト名を変更します。さらに、asadmin ユーティリティーコマンドのupdate-node-config またはupdate-node-ssh サブコマンドで、ノードの構成情報を更新します。

前提条件

- Application Server が起動している
- ドメイン管理サーバが起動している

想定ユーザー

- システム構築者

操作手順

1. asadmin ユーティリティーコマンドのstop-servers サブコマンドを実行して、パフォーマンストレーサー、サーバインスタンス、および Web サーバを一括停止します。

```
asadmin stop-servers
```

コマンドの実行結果を次に示します。

```
Command stop-servers executed successfully.
```

2. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがnot running になっていることを確認してください。

```
パフォーマンストレーサー名 not running
Command list-prfs executed successfully.
```

3. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがnot running になっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 not running
Command list-instances executed successfully.
```

- クラスター名は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。

4. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがnot running になっていることを確認してください。

```
Webサーバ名 not running
Command list-webservers executed successfully.
```

5. ドメイン管理サーバのホストの IP アドレスまたはホスト名を変更する場合は、次に示す手順を実施します。

リモートホストの IP アドレスまたはホスト名を変更する場合は、手順 6 の操作に進んでください。

a. asadmin ユーティリティーコマンドのstop-domain サブコマンドを実行して、ドメイン管理サーバを停止します。

```
asadmin stop-domain
```

コマンドの実行結果を次に示します。

```
Command stop-domain executed successfully.
```

b. ドメイン管理サーバのホストに設定されている IP アドレスまたはホスト名を、OS の手順に従って変更します。

必要に応じて、マシンを再起動してください。

c. asadmin ユーティリティーコマンドのstart-domain サブコマンドを実行して、ドメイン管理サーバを起動します。

```
asadmin start-domain
```

コマンドの実行結果を次に示します。

```
Command start-domain executed successfully.
```

d. asadmin ユーティリティーコマンドのlist-nodes サブコマンドを実行して、ノードの一覧を表示します。

```
asadmin list-nodes
```

コマンドの実行結果を次に示します。

```
ノード名 CONFIG IPアドレスまたはホスト名
ノード名 SSH IPアドレスまたはホスト名
Command list-nodes executed successfully.
```

e. ドメイン管理サーバと同じホストにノードがある場合は、asadmin ユーティリティーコマンドの update-node-config サブコマンドを実行して、ノードの構成情報を更新します。

デフォルトのノード（localhost-domain1）のようにローカルホストで登録されているノードの場合、この操作は必要ありません（ローカルホストでホスト名を解決できるため）。

```
asadmin update-node-config --nodehost 変更後のIPアドレスまたはホスト名 ノード名
```

コマンドの実行結果を次に示します。

```
Command update-node-config executed successfully.
```

f. リモートホストにノードがある場合（手順 d で確認したノードの一覧に、ノードタイプSSH のノードがある場合）は、リモートホストがドメイン管理サーバのホスト（ローカルホスト）のホスト名を解決できるようにします。

リモートホストの hosts ファイルを編集します。

g. リモートホストにノードがある場合は、リモートホストにある設定ファイル（das.properties）を修正します。

設定ファイル（das.properties）のパラメーター（agent.das.host）に、変更後の IP アドレスまたはホスト名を指定してください。設定ファイル（das.properties）は、*Java EE Serverのインストールディレクトリー*/glassfish/nodes/*ノード名*/agent/config/das.properties に格納されています。

6. リモートホストの IP アドレスまたはホスト名を変更する場合は、次に示す手順を実施します。

a. リモートホストに設定されている IP アドレスまたはホスト名を、OS の手順に従って変更します。

必要に応じて、マシンを再起動してください。

b. asadmin ユーティリティーコマンドのlist-nodes サブコマンドを実行して、ノードの一覧を表示します。

```
asadmin list-nodes
```

コマンドの実行結果を次に示します。

```
ノード名 CONFIG IPアドレスまたはホスト名
ノード名 SSH IPアドレスまたはホスト名
Command list-nodes executed successfully.
```

c. リモートホストのノード（手順 b で確認したノードの一覧で、ノードタイプSSH が表示されたノード）に対してasadmin ユーティリティーコマンドのupdate-node-ssh サブコマンドを実行し、ノードの構成情報を更新します。

```
asadmin --user ドメイン管理サーバのユーザー名 --passwordfile パスワードファイルのパス
update-node-ssh --nodehost 変更後のIPアドレスまたはホスト名 ノード名
```

コマンドの実行結果を次に示します。

```
Command update-node-ssh executed successfully.
```

7. asadmin ユーティリティーコマンドのstart-servers サブコマンドを実行して、Application Server を一括で起動します。

```
asadmin start-servers
```

コマンドの実行結果を次に示します。

```
Command start-servers executed successfully.
```

8. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがrunning になっていることを確認してください。

```
パフォーマンストレーサー名 running
Command list-prfs executed successfully.
```

9. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがrunning になっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 running
Command list-instances executed successfully.
```

- *クラスター名*は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。
- サーバインスタンスの起動に成功したかどうかは、Java EE サーバ（サーバインスタンス）のメッセージログに KDKD20031-I が出力されているかどうか、または Administration Console の Java EE サーバ（サーバインスタンス）のステータスでも確認できます。

10. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがrunning になっていることを確認してください。

```
Webサーバ名    running
Command list-webservers executed successfully.
```

8.5 環境情報をバックアップする

環境定義の変更に際して変更前後の設定を保管する場合などに、Application Server の環境情報をバックアップします。ドメインディレクトリーで管理しているファイルは、asadmin ユーティリティーコマンドのbackup-domain サブコマンドでバックアップファイルを作成します。 なお、ノードディレクトリーやサーバが使用するファイル（主にサーバテンプレート）をドメインディレクトリーの外に作成した場合は、手動でバックアップする必要があります。

前提条件

- ドメイン管理サーバが停止している
- Application Server が停止している

想定ユーザー

- システム構築者またはシステム運用者

操作手順

1. asadmin ユーティリティーコマンドのbackup-domain サブコマンドを実行して、ドメインに対応する Application Server の環境情報をバックアップします。

   ```
   asadmin backup-domain
   --backupdir バックアップファイルを格納するディレクトリーのパス ドメイン名
   ```

   コマンドの実行結果を次に示します。

   ```
   Command  backup-domain executed successfully.
   ```

2. backup-domain サブコマンドの実行時に指定したディレクトリーを参照し、バックアップファイルが格納されていることを確認します。

3. ドメインディレクトリー以外で管理しているファイル（主にサーバテンプレート）を、バックアップ用のディレクトリーにコピーしてバックアップします。

8.6　環境情報をリストアする

バックアップした Application Server の環境情報をリストア（復元）するには、asadmin ユーティリティーコマンドのrestore-domain サブコマンドで、バックアップファイルの環境情報をドメインディレクトリーに復元します。また、ドメインディレクトリー以外で管理するサーバテンプレートなどのファイルを、元の格納ディレクトリーにコピーして戻します。

前提条件

- ドメイン管理サーバが停止している
- Application Server が停止している

想定ユーザー

- システム構築者またはシステム運用者

操作手順

1. asadmin ユーティリティーコマンドのrestore-domain サブコマンドを実行して、バックアップファイルを基に、Application Server の環境情報を指定したドメインのドメインディレクトリーに復元します。

   ```
   asadmin restore-domain
   --backupdir バックアップファイルが格納されているディレクトリーのパス 環境情報をリストアするドメイン名
   ```

   コマンドの実行結果を次に示します。

   ```
   Command  restore-domain executed successfully.
   ```

2. restore-domain サブコマンドの実行時に指定したドメインのドメインディレクトリーを参照し、環境情報が復元されたことを確認します。

3. バックアップ用のディレクトリーに格納されているサーバテンプレートなどのファイルを、元の格納ディレクトリーにコピーして戻します。

8.7　修正パッチおよび修正版を適用する

Application Server の修正パッチが発行された際やシステムの定期メンテナンス時などに、修正パッチ・修正版をインストールしてシステムに適用します。修正パッチ・修正版を適用するには、稼働中の Application Server とドメイン管理サーバを停止させたあと、提供媒体を使用してインストールを実施します。

前提条件

- Application Server が起動している
- ドメイン管理サーバが起動している

想定ユーザー

- システム構築者

操作手順

1. Application Server の修正パッチおよび修正版の提供媒体を入手します。

   ソフトウェアサポートサービスの Web サイトまたは修正パッチ CD-ROM から修正パッチがリリースされているか確認し、リリースされている場合は、そのどちらかから提供媒体を入手してください。

2. asadmin ユーティリティーコマンドのstop-servers サブコマンドを実行して、パフォーマンストレーサー、サーバインスタンス、および Web サーバを一括停止します。

   ```
   asadmin stop-servers
   ```

   コマンドの実行結果を次に示します。

   ```
   Command stop-servers executed successfully.
   ```

3. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

   ```
   asadmin list-prfs
   ```

   コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがnot running になっていることを確認してください。

   ```
   パフォーマンストレーサー名 not running
   Command list-prfs executed successfully.
   ```

4. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

   ```
   asadmin list-instances --long=true
   ```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがnot runningになっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 not running
Command list-instances executed successfully.
```

- クラスター名は、複数のJava EEサーバを配置するクラスター構成の場合だけ表示されます。

5. asadminユーティリティーコマンドのlist-webserversサブコマンドを実行して、Webサーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Webサーバのステータスがnot runningになっていることを確認してください。

```
Webサーバ名 not running
Command list-webservers executed successfully.
```

6. asadminユーティリティーコマンドのstop-domainサブコマンドを実行して、ドメイン管理サーバを停止します。

```
asadmin stop-domain
```

コマンドの実行結果を次に示します。

```
Command stop-domain executed successfully.
```

7. 提供媒体を使用して修正パッチおよび修正版をインストールします。

適用手順については、修正パッチに添付されているRELEASE.TXTを参照してください。

8. 修正パッチおよび修正版がインストールされたことを［セットアップの完了］ダイアログで確認します。

9. asadminユーティリティーコマンドのstart-domainサブコマンドを実行して、ドメイン管理サーバを起動します。

```
asadmin start-domain
```

コマンドの実行結果を次に示します。

```
Command start-domain executed successfully.
```

10. asadminユーティリティーコマンドのstart-serversサブコマンドを実行して、Application Serverを一括で起動します。

```
asadmin start-servers
```

コマンドの実行結果を次に示します。

```
Command start-servers executed successfully.
```

11. asadmin ユーティリティーコマンドのlist-prfs サブコマンドを実行して、パフォーマンストレーサーの一覧を表示します。

```
asadmin list-prfs
```

コマンドの実行結果を次に示します。パフォーマンストレーサーのステータスがrunning になっていることを確認してください。

```
パフォーマンストレーサー名 running
Command list-prfs executed successfully.
```

12. asadmin ユーティリティーコマンドのlist-instances サブコマンドに--long オプションを指定して実行し、サーバインスタンスの一覧を表示します。

```
asadmin list-instances --long=true
```

コマンドの実行結果を次に示します。サーバインスタンスのステータスがrunning になっていることを確認してください。

```
サーバインスタンス名 ホスト名 ポート番号 プロセスID クラスター名 running
Command list-instances executed successfully.
```

- クラスター名は、複数の Java EE サーバを配置するクラスター構成の場合だけ表示されます。
- サーバインスタンスの起動に成功したかどうかは、Java EE サーバ（サーバインスタンス）のメッセージログに KDKD20031-I が出力されているかどうか、または Administration Console の Java EE サーバ（サーバインスタンス）のステータスでも確認できます。

13. asadmin ユーティリティーコマンドのlist-webservers サブコマンドを実行して、Web サーバの一覧を表示します。

```
asadmin list-webservers
```

コマンドの実行結果を次に示します。Web サーバのステータスがrunning になっていることを確認してください。

```
Webサーバ名    running
Command list-webservers executed successfully.
```

関連項目

- 8.8　修正パッチ適用時のエラーメッセージ

8.8 修正パッチ適用時のエラーメッセージ

UNIX の場合に、修正パッチ適用時の障害を通知するメッセージについて、意味と対処を説明します。

エラーメッセージの意味と対処

修正パッチ適用時、標準エラー出力に出力されるメッセージについて、意味と対処を説明します。

メッセージ ID	メッセージ	意味	対処
KESP5100-E	usage : %s [-D *log_file*]	UPDATE の引数に誤りがあります。	引数を確認して、再度実行してください。
KESP5101-E	An attempt to execute the pre-processing program for the update has failed. (details code = %d) Read the product documentation and follow the instructions.	前処理に失敗しました。	弊社問い合わせ窓口へ連絡してください。
KESP5102-E	An attempt to execute the post-processing program for the update has failed. (details code = %d) Read the product documentation and follow the instructions.	後処理に失敗しました。	弊社問い合わせ窓口へ連絡してください。
KESP5103-E	The module replacement program has failed. (details code = %d) Read the product documentation and follow the instructions.	モジュールの入れ替えに失敗しました。	弊社問い合わせ窓口へ連絡してください。
KESP5104-E	An attempt to execute the patch program for the update has failed. (details code = %d)	MODPATCH が失敗しました。	ログファイルを確認してください。

メッセージID	メッセージ	意味	対処
KESP5105-E	You must be a superuser to run this program.	ルート権限のないユーザーで実行したため処理に失敗しました。	ルート権限のあるユーザーが実行してください。
KESP5106-E	This patch is for the %s platform. The patch cannot be applied to the %s platform.	対象外のプラットフォームで実行されました。	対象プラットフォームとパッチの製品情報ファイル（PRODUCT.INI）の[! comfirm]platform を確認してください。
KESP5107-E	Sufficient memory could not be allocated. Please retry the operation.	メモリが不足しています。	修正パッチ適用に必要なメモリを確保してから再度実行してください。
KESP5108-E	Disk space is insufficient. Increase the available disk space, and then retry the operation.	ディスクの空き容量が不足しています。	修正パッチ適用に必要な空き容量を確保してから再度実行してください。
KESP5109-E	An attempt to modify the file pplistd(%s), has failed. Please retry the operation.	インストール製品情報ファイル（pplistd）の更新中に解析に失敗しました。	インストール製品情報ファイル（pplistd）のフォーマットが正しいか確認してください。
KESP5110-E	The file (%s) was not found. Get the patch file again.	修正パッチに必要なファイルが存在しません。	表示されたファイルが修正パッチにアーカイブされていることを確認してください。
KESP5111-E	The file pplistd was not found. (%s)	インストール製品情報ファイル（pplistd）が存在しません。	インストール製品情報ファイル（pplistd）が存在するか確認してください。
KESP5112-E	The installation path does not exist. (%s)	インストールディレクトリが存在しません。	インストールディレクトリが存在しているか確認してください。
KESP5113-E	The check program for the update has failed. (details code = %d)	チェッカープログラムの実行に失敗しました。	デバッグモードで再度実行して、デバッグログを確認してください。
KESP5114-E	An attempt to modify the history file has failed. (%s)	履歴ファイルの更新に失敗しました。	表示されたファイル名と同じディレクトリが存在しないことを確認してください。
KESP5115-E	The format of the file (%s) is invalid.	製品情報ファイル（PRODUCT.INI）、またはインストール製品情報ファイル（pplistd）のフォーマットに誤りがあります。	製品情報ファイル（PRODUCT.INI）、またはインストール製品情報ファイル（pplistd）のフォーマットを確認してください。

メッセージID	メッセージ	意味	対処
KESP5116-E	[%s] %s is too long.	製品情報ファイル（PRODUCT.INI）に定義された値が指定できる最大長を超えています。	製品情報ファイル（PRODUCT.INI）に定義された値を確認してください。
KESP5117-E	[%s] %s has not been written.	製品情報ファイル（PRODUCT.INI）に必要なエントリーが存在しません。	製品情報ファイル（PRODUCT.INI）に必要なエントリーが記述されていることを確認してください。
KESP5118-E	The specified platform key in the [!confirm] section is invalid.	製品情報ファイル（PRODUCT.INI）に定義された platform の値に誤りがあります。	製品情報ファイル（PRODUCT.INI）に定義された platform の値が、HPUX、SOLARIS、DIGITAL、LINUX、AIX、IPLINUX、IPFHPUX、X64LIN のどれかであることを確認してください。
KESP5119-E	The specified installtype key in the [!common] section is invalid.	製品情報ファイル（PRODUCT.INI）に定義された installtype の値に誤りがあります。	製品情報ファイル（PRODUCT.INI）に定義された installtype の値が、HIPPINST であることを確認してください。
KESP5120-E	The version of the file (%s) is not 1.0.	ファイルバージョンが 1.0 以外です。	ファイルバージョンが 1.0 であることを確認してください。
KESP5121-E	The installtype (%s) is not "%s".	インストールタイプに誤りがあります。	インストールタイプが、HIPPINST であることを確認してください。
KESP5122-E	The file PRODUCT.INI is invalid.	製品情報ファイル（PRODUCT.INI）に誤りがあります。	製品情報ファイル（PRODUCT.INI）のファイルフォーマットを確認してください。
KESP5123-E	The file (%s) cannot be opened.	ファイルを開けませんでした。	表示されたファイルが使用中でないことを確認してください。また、ディスクの空き容量を確認してください。
KESP5124-E	The %s processing has failed.	表示された処理に失敗しました。	ログファイルを確認してください。
KESP5125-E	The %s system call has failed.	表示されたシステムコールの実行に失敗しました。	ログファイルを確認してください。
KESP5126-E	The specified logfilemode key in the [!updater] section is invalid.	製品情報ファイル（PRODUCT.INI）に定義された logfilemode の値に誤りがあります。	製品情報ファイル（PRODUCT.INI）に定義された logfilemode の値がOVERWRITE、またはAPPEND であることを確認してください。
KESP5129-E	The specified unpatch key in the [!modpatch] section is invalid.	製品情報ファイル（PRODUCT.INI）に定義された unpatch の値に誤りがあります。	製品情報ファイル（PRODUCT.INI）に定義された unpatch の値がSPKUTIL.TAR であることを確認してください。
KESP5130-E	The specified unpatchversion key in the [!modpatch] section is invalid.	製品情報ファイル（PRODUCT.INI）に定義された unpatchversion の値に誤りがあります。	製品情報ファイル（PRODUCT.INI）に定義された unpatchversion の値が0303 であることを確認してください。

メッセージID	メッセージ	意味	対処
KESP5131-E	[%s]%s is invalid.	製品情報ファイル（PRODUCT.INI）に定義されたパス定義に誤りがあります。	製品情報ファイル（PRODUCT.INI）のエントリーに定義されたパスが正しいことを確認してください。
KESP5132-E	The installation stopped because the version information of the selected product has been changed and is incorrect.	指定された製品のバージョン情報が正しくありません。	CD-ROM 媒体に問題があります。弊社問い合わせ窓口へ連絡してください。
KESP5133-E	The update stopped because the version information of the installed product has been changed and is incorrect.	インストールされている製品のバージョン情報が正しくありません。	インストールされている製品に問題があります。弊社問い合わせ窓口へ連絡してください。
KESP5134-E	An attempt to start %s failed.	アップデート要否チェックの起動に失敗しました。	アップデート要否チェック DLL が正しい DLL であることを確認してください。
KESP5135-E	The rollback has failed. Please rerun UPDATE.	ロールバックに失敗しました。	UPDATE を再実行してください。
KESP5255-E	A fatal error occurred.	致命的なエラーが発生しました。	ログファイルを確認してください。

8.9 システムの利用状況を確認する

マシン構成や環境定義の見直しなどに向けて、アクセスログを参照し、システムの利用状況を示す業務分布やリクエスト数の情報を確認および現状分析します。

前提条件

- Application Server がセットアップされている
- サーバインスタンスでアプリケーションが稼働している
- 業務システムが稼働し、Web クライアントから送信された HTTP リクエストを受け付けている

想定ユーザー

- システム構築者

操作手順

1. アクセスログを参照して、システムの利用状況を確認します。

   参照するファイルを次に示します。ファイル名の *x* には可変値が付与されます。

   - 業務分布（ファイル名：*Application Serverのインストールディレクトリー*/javaee/logs/nodes/*ノード名*/*Webサーバ名*/access.*x*）
   - 1 秒当たりのリクエスト数（ファイル名：*Application Serverのインストールディレクトリー*/javaee/logs/nodes/*ノード名*/*Webサーバ名*/access.*x*）

   なお、このファイルで確認できる 1 秒当たりのリクエスト数には、Java EE サーバで処理されるリクエストのほかに、Web サーバで処理される静的コンテンツへのリクエストも含まれています。この点を考慮して、利用状況の確認や分析を実施してください。

8.10　システムの動作状況を確認する

マシン構成や環境定義の見直しなどに向けて現状分析するため、稼働情報ファイルを参照し、システムの動作状況を確認します。稼働情報ファイルには、Java EE サーバを対象に、Application Server の稼働情報収集機能によって定期的に収集されたシステムの稼働情報が出力されます。このファイルを基に、システムの動作状況を示す情報を確認、分析します。

前提条件

- Application Server がセットアップされている
- サーバインスタンスでアプリケーションが稼働している

想定ユーザー

- システム構築者

操作手順

1. 稼働情報ファイルの出力先ディレクトリーから、確認対象の期間に該当するファイルを取得します。

   デフォルトの出力先ディレクトリーは、*Application Serverのインストールディレクトリー*/javaee/logs/nodes/*ノード名*/*サーバインスタンス名*/statistics です。

2. 取得した稼働情報ファイルを参照し、システムの動作状況を確認します。

   参照する稼働情報ファイルと確認項目を次に示します。ファイル名の*YYYYMMDDhhmm* にはファイル出力時の日付と時刻が、*TZ* には GMT からの数値タイムゾーンオフセット（日本標準時の場合は+0900）が付与されます。

   - Java VM メモリ拡張稼働情報ファイル（ファイル名：JVMMemoryExtensionsStatistics_*サーバインスタンス名*_*YYYYMMDDhhmmTZ*.csv）

確認項目	属性名	説明
ThreadBlockedCount	ThreadBlockedCount.Count	Java VM 上でモニターロックのためにブロック状態にあるスレッド数
EdenUsedMemorySize	EdenUsedMemorySize.Count	現在使用されている Eden 領域のメモリーサイズ
EdenTotalMemorySize	EdenTotalMemorySize.Count	使用できることが保証（コミット）されている Eden 領域のメモリーサイズ
EdenMaxMemorySize	EdenMaxMemorySize.Count	使用できる Eden 領域の最大メモリーサイズ
SurvivorUsedMemorySize	SurvivorUsedMemorySize.Count	現在使用されている Survivor 領域のメモリーサイズ
SurvivorTotalMemorySize	SurvivorTotalMemorySize.Count	使用できることが保証（コミット）されている Survivor 領域のメモリーサイズ
SurvivorMaxMemorySize	SurvivorMaxMemorySize.Count	使用できる Survivor 領域の最大メモリーサイズ

確認項目	属性名	説明
TenuredUsedMemorySize	TenuredUsedMemorySize.Count	現在使用されている Tenured 領域のメモリーサイズ
TenuredTotalMemorySize	TenuredTotalMemorySize.Count	使用できることが保証（コミット）されている Tenured 領域のメモリーサイズ
TenuredMaxMemorySize	TenuredMaxMemorySize.Count	使用できる Tenured 領域の最大メモリーサイズ
MetaspaceUsedMemorySize	MetaspaceUsedMemorySize.Count	現在使用されている Metaspace 領域のメモリーサイズ
MetaspaceTotalMemorySize	MetaspaceTotalMemorySize.Count	使用できることが保証（コミット）されている Metaspace 領域のメモリーサイズ
MetaspaceMaxMemorySize	MetaspaceMaxMemorySize.Count	使用できる Metaspace 領域の最大メモリーサイズ
ExplicitHeapSize	ExplicitHeapSize.Count	Explicit ヒープサイズ
ExplicitMemoryBlockCount	ExplicitMemoryBlockCount.Count	Explicit ヒープ領域の Explicit メモリーブロックの個数
ExplicitMemoryBlockMaxSize	ExplicitMemoryBlockMaxSize.Count	Explicit メモリブロックの最大サイズ
HttpSessionExplicitMemoryBlockMaxSize	HttpSessionExplicitMemoryBlockMaxSize.Count	HTTP セッションで取得した Explicit メモリーブロックの最大サイズ
HttpSessionExplicitMemoryBlockCount	HttpSessionExplicitMemoryBlockCount.Count	HTTP セッションで取得した Explicit メモリーブロックの個数
ContainerExplicitHeapSize	ContainerExplicitHeapSize.Count	HTTP セッションで取得した Explicit ヒープ領域を除く、コンテナーが管理している Explicit ヒープサイズ
ApplicationExplicitHeapSize	ApplicationExplicitHeapSize.Count	アプリケーションおよび Java VM が管理している Explicit ヒープサイズ

- ネットワークコネクションキュー稼働情報ファイル（ファイル名：NetworkConnectionQueueStatistics_*サーバインスタンス名*_*YYYYMMDDhhmmTZ*.csv）

確認項目	属性名	説明
CountOverflows	CountOverflows.Count	コネクションキューからあふれた数（累積値）
ExCountQueued	－	キュー内にあるコネクション数
	ExCountQueued.Current	コネクションキューの現在値
	ExCountQueued.HighWaterMark	前回の稼働情報の出力時点からのコネクションキューに格納したコネクションの最大値
	ExCountQueued.LowWaterMark	前回の稼働情報の出力時点からのコネクションキューに格納したコネクションの最小値

（凡例）－：該当しない。

- ネットワークスレッドプール稼働情報ファイル（ファイル名：NetworkThreadPoolStatistics_*サーバインスタンス名_YYYYMMDDhhmmTZ*.csv）

確認項目	属性名	説明
ExCurrentThreadsBusy	－	リクエストを処理するリスナースレッドプール内で使用しているリクエスト処理スレッドの数
	ExCurrentThreadsBusy.Current	実行中のスレッドの現在値
	ExCurrentThreadsBusy.HighWaterMark	前回の稼働情報の出力時点からの同時に実行したスレッドの最大値
	ExCurrentThreadsBusy.LowWaterMark	前回の稼働情報の出力時点からの同時に実行したスレッドの最小値

（凡例）－：該当しない。

- Web セッション稼働情報ファイル（ファイル名：WebSessionStatistics_*サーバインスタンス名_YYYYMMDDhhmmTZ*.csv）

確認項目	属性名	説明
ActiveSessions	－	アクティブなセッションの数
	ActiveSessions.Current	セッション数の現在値
	ActiveSessions.HighWaterMark	前回の稼働情報の出力時点からのセッション数の最大値
	ActiveSessions.LowWaterMark	前回の稼働情報の出力時点からのセッション数の最小値

（凡例）－：該当しない。

- JDBC 接続プール稼働情報ファイル（ファイル名：JDBCConnectionPoolStatistics_*サーバインスタンス名_YYYYMMDDhhmmTZ*.csv）

確認項目	属性名	説明
NumConnUsed	－	使用されているコネクションの合計数
	NumConnUsed.Current	使用中コネクションの現在値
	NumConnUsed.HighWaterMark	前回の稼働情報の出力時点からの使用中となったプール内コネクションの最大値
	NumConnUsed.LowWaterMark	前回の稼働情報の出力時点からの使用中となったプール内コネクションの最小値
NumConnFree	－	最後のサンプリング時点におけるプール内コネクションの合計数
	NumConnFree.Current	未使用コネクションの現在値

確認項目	属性名	説明
	NumConnFree.HighWaterMark	前回の稼働情報の出力時点からの未使用となったプール内コネクションの最大値
	NumConnFree.LowWaterMark	前回の稼働情報の出力時点からの未使用となったプール内コネクションの最小値

（凡例）－：該当しない。

8.11　システムをスケールアウトする

業務処理量の増加が判明した場合や、将来予測される業務処理量に備える場合などに、増設したマシン（新設サーバ）に Application Server を構築して、システムをスケールアウトします。システムをスケールアウトするには、新設サーバに Application Server をインストールし、新設サーバのローカルホストから既存サーバのリモートホストへ接続するための設定をしたあと、Application Server のセットアップを実施します。

前提条件

- クラスター構成で、既存のサーバインスタンスがクラスターに含まれている
- 既存サーバでドメイン管理サーバが起動している
- 既存サーバに Application Server がセットアップされている
- 新設サーバが稼働している

想定ユーザー

- システム構築者

操作手順

1. 新設サーバに Application Server をインストールします。
2. リモートホストで、ドメイン管理サーバ（ローカルホスト）のホスト名を名前解決できるように hosts ファイルを編集します。
3. ローカルホストで、パスワードファイルを作成します。

```
AS_ADMIN_SSHPASSWORD=リモートホストのパスワード
```

4. ローカルホストで、asadmin ユーティリティーコマンドのsetup-ssh サブコマンドを実行して、暗号キー（SSL キー）を作成します。

```
asadmin --passwordfile パスワードファイルのパス
setup-ssh --sshuser リモートホストのSSHユーザー名 リモートホスト名
```

コマンドの実行結果を次に示します。

```
Command setup-ssh executed successfully.
```

5. ローカルホストで、asadmin ユーティリティーコマンドのcreate-password-alias サブコマンドを実行して、パスワードに対するエイリアスを設定します。

```
asadmin create-password-alias パスワードエイリアス名
```

パスワードが求められた場合は、リモートホストのパスワードを入力してください。

コマンドの実行結果を次に示します。

```
Command create-password-alias executed successfully.
```

6. ローカルホストで、手順 3 で作成したパスワードファイルを編集します。

```
AS_ADMIN_SSHPASSWORD=${ALIAS=パスワードエイリアス名}
```

7. ドメイン管理サーバを再起動します。

```
asadmin restart-domain
```

コマンドの実行結果を次に示します。

```
Command restart-domain executed successfully.
```

8. 新設サーバに Application Server をセットアップするため、asadmin ユーティリティーコマンドの create-node-ssh サブコマンドを実行して、リモートホストに対するノードを追加します。

```
asadmin --user ドメイン管理サーバのユーザー名
--passwordfile パスワードファイルのパス
create-node-ssh --nodehost リモートホストのホスト名
--sshuser リモートホストのアカウント名 --sshkeyfile ~/.ssh/id_rsa
--installdir Application Serverのインストールディレクトリー/javaeeの絶対パス リモートホストのノード名
```

*リモートホストのノード名*には、リモートホストに構築するノード名（ノードを識別するための任意の名称）を指定します。

コマンドの実行結果を次に示します。

```
Command create-node-ssh executed successfully.
```

9. 新設サーバに Application Server をセットアップするため、asadmin ユーティリティーコマンドの create-prf サブコマンドを実行して、パフォーマンストレーサーを構築します。

```
asadmin create-prf --node リモートホストのノード名 パフォーマンストレーサー名
```

コマンドの実行結果を次に示します。

```
Command create-prf executed successfully.
```

10. 新設サーバに Application Server をセットアップするため、asadmin ユーティリティーコマンドの create-instance サブコマンドを実行して、Java EE サーバ（サーバインスタンス）を構築します。

```
asadmin create-instance --node リモートホストのノード名
--prf パフォーマンストレーサー名
--cluster クラスター名 サーバインスタンス名
```

*クラスター名*には、構築済みのクラスター名を指定します。

コマンドの実行結果を次に示します。

```
Command create-instance executed successfully.
```

11. ハードウェアロードバランサーを使用している場合は、asadmin ユーティリティーコマンドのcreate-webserver サブコマンドを実行して、新設サーバに Web サーバを構築します。

ソフトウェアロードバランサーを使用している場合は、手順 15 の操作に進んでください。

```
asadmin create-webserver --node リモートホストのノード名 Webサーバ名
```

コマンドの実行結果を次に示します。

```
Command create-webserver executed successfully.
```

12. ハードウェアロードバランサーを使用している場合は、asadmin ユーティリティーコマンドのcreate-relation サブコマンドを実行して、新設サーバにサーバ間関連を設定します。

```
asadmin create-relation --relationtype redirect
--from Webサーバ名
--to サーバインスタンス名
--properties サーバ間関連のプロパティー名=サーバ間関連のプロパティーの値 サーバ間関連の関連名
```

リダイレクト関連の関連づけを設定する場合、静的コンテンツを Web サーバで処理し、静的コンテンツ以外のリクエストを Java EE サーバで処理するために、--properties オプションにpath、およびnetwork-listener を指定します。

(例)

path=/apserver/:network-listener=http-listener-1

path には、スラッシュ（/）から始まる URL のパスを指定します。スラッシュだけの指定（path=/）はしないでください。この例では、http://xxxxxxxxxx/index.html のように URL のパスの先頭にapserver を含まないリクエストは、Web サーバの静的コンテンツにアクセスされます。また、http://xxxxxxxxxx/apserver/sample/index.jsp のように URL のパスの先頭にapserver を含むリクエストは、http://yyyyyyyyyy/sample/index.jsp という URL で Java EE サーバにリダイレクトされます。

network-listener には、リダイレクト先の Java EE サーバの HTTP リスナー、または HTTPS リスナーを示すネットワークリスナー名を指定します。Java EE サーバには、デフォルトで HTTP リスナーにhttp-listener-1、HTTPS リスナーにhttp-listener-2 というネットワークリスナーが定義されています。

コマンドの実行結果を次に示します。

```
Command create-relation executed successfully.
```

13. ハードウェアロードバランサーを使用している場合で、かつ、性能を考慮して静的コンテンツを Web サーバに配置する場合は、Web サーバのドキュメントルートディレクトリーに静的コンテンツを格納します。

Web サーバのドキュメントルートディレクトリーは、*Application Server*インストールディレクトリー/javaee/glassfish/nodes/*ノード名*/*Webサーバ名*/root/htdocs です。

14. ハードウェアロードバランサーを使用している場合は、ハードウェアロードバランサーに対してリクエストの振り分けを設定します。

リクエストを振り分ける設定については、使用しているハードウェアロードバランサーのマニュアルを参照してください。

15. ソフトウェアロードバランサーを使用している場合は、asadmin ユーティリティーコマンドのcreate-relation サブコマンドを実行して、新設サーバにサーバ間関連を設定します。

```
asadmin create-relation --relationtype redirect
--from Webサーバ名
--to クラスター名
--properties サーバ間関連のプロパティー名=サーバ間関連のプロパティーの値  サーバ間関連の関連名
```

リダイレクト関連の関連づけを設定する場合、静的コンテンツを Web サーバで処理し、静的コンテンツ以外のリクエストを Java EE サーバで処理するために、--properties オプションにpath、および network-listener を指定します。

(例)

```
path=/apserver/:network-listener=http-listener-1
```

path には、スラッシュ（/）から始まる URL のパスを指定します。スラッシュだけの指定（path=/）はしないでください。この例では、http://xxxxxxxxxxx/index.html のように URL のパスの先頭に apserver を含まないリクエストは、Web サーバの静的コンテンツにアクセスされます。また、http://xxxxxxxxxxx/apserver/sample/index.jsp のように URL のパスの先頭にapserver を含むリクエストは、http://yyyyyyyyyyy/sample/index.jsp という URL で Java EE サーバにリダイレクトされます。

network-listener には、リダイレクト先の Java EE サーバの HTTP リスナー、または HTTPS リスナーを示すネットワークリスナー名を指定します。Java EE サーバには、デフォルトで HTTP リスナーにhttp-listener-1、HTTPS リスナーにhttp-listener-2 というネットワークリスナーが定義されています。

コマンドの実行結果を次に示します。

```
Command create-relation executed successfully.
```

8.12 Application Server をバージョンアップする

Application Server をバージョンアップするには、旧バージョンのドメインを新バージョンに適用するためのアップグレードが必要です。 この作業には、asadmin ユーティリティーコマンドのbackup-domain サブコマンドで、旧バージョンのドメインのバックアップファイルを作成し、新バージョンをインストールしたあとに、asadmin ユーティリティーコマンドのrestore-domain サブコマンドで、ドメインを新環境にリストアします。このリストアしたドメインに対してasadmin ユーティリティーコマンドのstart-domain サブコマンドを実行して、ドメイン管理サーバの構成をアップグレードします 。

前提条件

- 旧バージョンのドメインが存在する

想定ユーザー

- システム構築者またはシステム運用者

操作手順

1. 旧バージョンの環境情報のバックアップを取得します。
   a. 旧バージョンの環境で稼働中のすべてのサーバおよびすべてのドメイン管理サーバを停止します。
   b. asadmin ユーティリティーコマンドのbackup-domain サブコマンドを実行して、旧バージョンの環境情報のバックアップファイルを出力します。

```
asadmin backup-domain
--backupdir バックアップファイルの格納ディレクトリー ドメイン名
```

   コマンドの実行結果を次に示します。

```
Command  backup-domain executed successfully.
```

   c. 次に示すバックアップファイルが出力されたことを確認します。

```
バックアップファイルの格納ディレクトリー/ドメイン名/ドメイン名_YYYY_MM_DD_v通番.zip
```

   通番は、00001 から始まる数値が割り当てられます。

2. 新バージョンをインストールします。
   a. 旧バージョン環境で稼働中のすべてのサーバおよびすべてのドメイン管理サーバを停止します。
   b. 新バージョンをインストールします。

   インストールの種類を次に示します。

   新規インストール：旧バージョンがインストールされているマシンとは別のマシンに、新バージョンをインストールする

   複数インストール：旧バージョンがインストールされているマシンで、別のディレクトリーに新バージョンをインストールする

上書きインストール：旧バージョンがインストールされているマシンで、同じディレクトリーに新バージョンをインストールする

複数の Java EE サーバを配置するクラスター構成にする場合は、リモートホストごとにインストールを実行します。

3. 旧バージョンでバックアップを取得した環境情報を、新バージョンの環境に移行します。

上書きインストールをした場合、この作業は不要です。

a. 旧バージョンの Java EE Server の環境変数定義ファイル（asenv.conf）に追加した環境変数を、新バージョンの同じファイルに設定します。

Java EE Server の環境変数定義ファイル

インストールディレクトリー/javaee/glassfish/config/asenv.conf

インストールディレクトリーには、Application Server のインストールディレクトリーを指定します。

b. 新バージョンの環境で、asadmin ユーティリティーコマンドのrestore-domain サブコマンドを実行して、バックアップした旧バージョンの環境情報を新バージョンの環境にリストア（復元）します。

```
asadmin restore-domain
--backupdir バックアップファイルの格納ディレクトリー 環境情報をリストアするドメイン名
```

c. リストアしたドメインのドメインディレクトリーを参照して、環境情報が復元されたことを確認します。

4. ドメインのアップグレードをします。

a. 新バージョンの環境で、asadmin ユーティリティーコマンドのstart-domain サブコマンドに--upgrade オプションを指定して実行し、ドメインのアップグレードをします。

```
asadmin start-domain --upgrade ドメイン名
```

コマンドの実行結果を次に示します。

```
Command start-domain executed successfully.
The DAS was stopped.
```

このとき、コンソールを参照して、 SEVERE、 ALERT、または EMERGENCY レベルのメッセージが出力されていないことを確認します。

b. 新バージョンの環境で、ドメインディレクトリー以下に出力された osgi-cache-*数字*という名称のディレクトリーを削除します。

ドメインのアップグレードを実行すると、ドメインディレクトリー以下にあった OSGi キャッシュディレクトリーがosgi-cache-*数字*という名前にリネームされます。アップグレード後は使用しないディレクトリーであるため、削除します。

c. asadmin ユーティリティーコマンドのstart-domain サブコマンドを実行して、ドメイン管理サーバを起動します。

```
asadmin start-domain ドメイン名
```

コマンドの実行結果を次に示します。

```
Command start-domain executed successfully.
```

d. asadmin ユーティリティーコマンドのupdate-node-config サブコマンドまたはupdate-node-ssh サブコマンドを実行して、各ノードのインストールディレクトリーを新バージョンのインストールディレクトリーに変更します。

ローカルホストのノードの場合

```
asadmin update-node-config
--installdir 新バージョンのインストールディレクトリー/javaeeの絶対パス
--nodedir ノードディレクトリー ノード名
```

SSH 接続のノードの場合

```
asadmin update-node-ssh --installdir 新バージョンのインストールディレクトリー/javaeeの
絶対パス  --nodedir
ノードディレクトリー ノード名
```

旧バージョンの環境でノードディレクトリーをデフォルトから変更していた場合は、--nodedir オプションを指定して、ノードディレクトリーも変更します。

ノードディレクトリーの変更が必要な場合は、ノードディレクトリーも変更してください。

コマンドの実行結果を次に示します。

ローカルホストのノードの場合

```
Command update-node-config executed successfully.
```

SSH 接続のノードの場合

```
Command update-node-ssh executed successfully.
```

5. 動作確認をします。

a. ドメイン上のサーバを起動し、サーバの動作確認とアプリケーションのテストを実施します。

b. テストで問題がないことを確認したら、新バージョンの本番環境として運用を開始します。

6. 必要に応じて、旧バージョンの環境を削除します。

上書きインストールした場合は、この手順は不要です。

旧バージョンの環境を複数の Java EE サーバを配置するクラスター構成にしていた場合、リモートホストごとに旧バージョンの環境をアンインストールします。

9 トラブルシュート資料の活用

アプリケーションの実行環境および開発環境では、トラブルシュート資料として各種ログ、トレースなどが出力されます。出力されるトラブルシュート資料の見方やパフォーマンストレーサーのトレース取得ポイントなど、トラブルシュート資料を活用する際に必要な情報について説明します。

9.1 Application Server が出力するトラブルシュート資料について

Application Server が出力するトラブルシュート資料には、メッセージログや性能解析トレースのほか、各プロセスの特徴に応じたトラブルシュート資料などがあります。必要に応じて、システム情報収集機能または手動でトラブルシュート資料を収集します。

システム情報収集機能とは

システム情報収集機能は、Application Server を使用して構築されたシステムに関する情報を一括収集する機能です。システム情報収集機能を障害発生時に用いることで、現象発生時点のダンプや設定ファイル・ログ・トレースといった障害の原因究明に必要な資料一式を収集することができます。

収集が必要なトラブルシュート資料

Application Server のインストール時およびシステムの運用時に障害が発生した場合に必要な、トラブルシュート資料を次の表に示します。なお、トラブルシュート資料は、システム情報収集機能を利用して収集します。システム情報収集機能で収集できない情報については、手動で収集する必要があります。

項番	障害発生のタイミング	トラブルシュート資料の内容	システム情報収集機能での収集可否
1	Application Server のインストール時	インストールログ	-
2	Application Server のシステム構築時、およびシステム運用時	各プロセスのログ・トレース	○
3		メモリーダンプ	○
4		定義・設定情報 （製品のバージョン情報、製品の設定ファイルなど ）	○
5		作業ディレクトリーの内容	×
6		画面のキャプチャー	×
7		標準エラー出力（コマンド実行時のコンソール出力）	×
8		性能解析トレース（CSV 形式）	○
9		OS のログ （syslog）	○
10		OS のログ （イベントログ）	×
11		OS の統計情報 (CPU 利用率、メモリー使用量、スレッド数など)	×
12		OS の状態情報 （環境変数、netstat/ps/sar などの各種コマンドの結果 ）	○

（凡例）

○：システム情報収集機能で収集できます。

×：システム情報収集機能で収集できません。

Application Server が出力するトラブルシュート資料の分類

Application Server を使用したシステムで障害が発生した場合に、Application Server が出力するトラブルシュート資料には、次のような種類があります。必要に応じて、トラブルシュート資料を収集して、調査してください。

なお、トラブルシュート資料の出力先ディレクトリーは、一部の資料を除きHJES_LOGSDIR 環境変数で変更できます。HJES_LOGSDIR 環境変数のデフォルト値は、"*Application Serverのインストールディレクトリー*/javaee/logs"です。

メッセージログ

Java EE サーバや Web サーバから出力されるログです。障害の原因や稼働状況が確認できます。

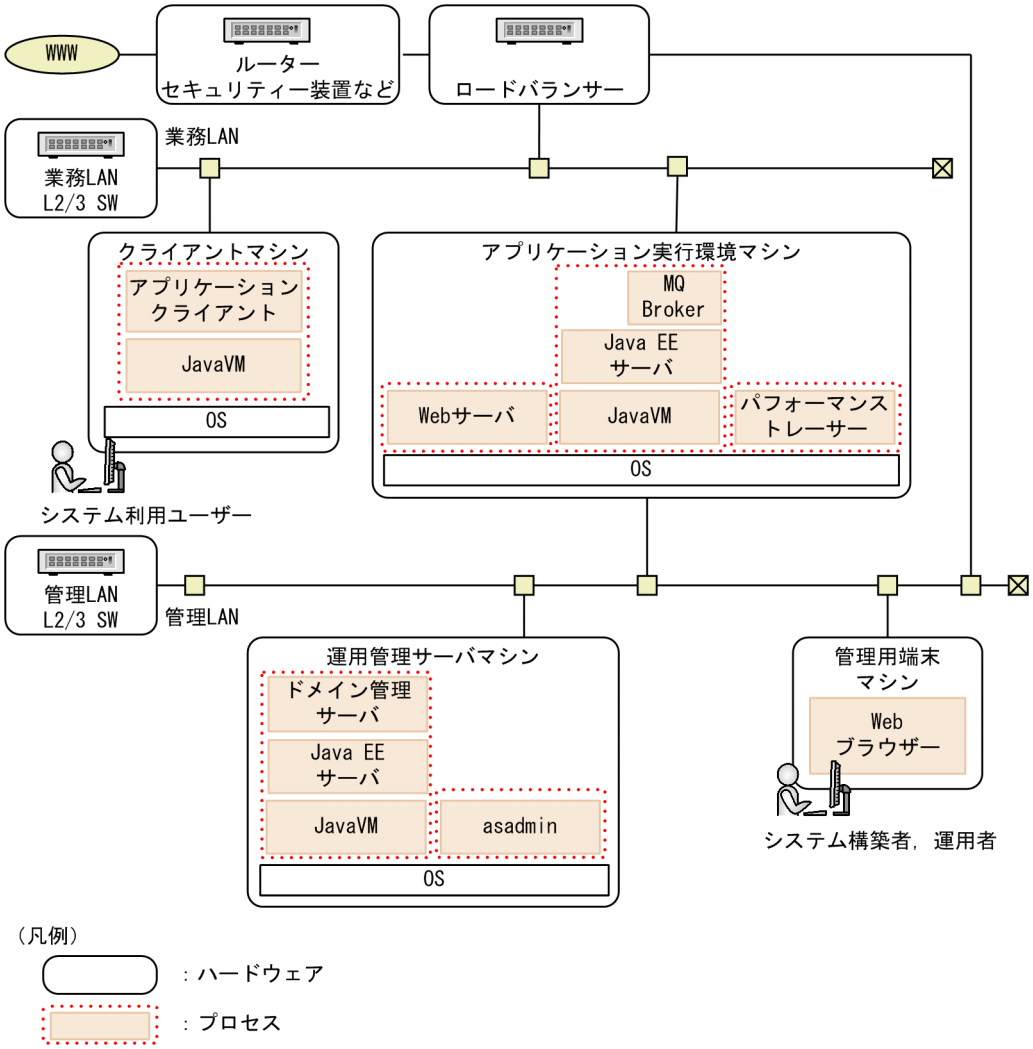
性能解析トレース

複数のプロセスにわたるリクエスト処理の流れを追跡できるトレース情報です。障害の解析や性能の解析に利用できます。

Java EE サーバおよび HiRDB で出力される ID 付きの性能解析トレースを、Web サーバのリクエストログに出力されるリクエストを識別する ID と突き合わせることで、リクエスト処理の追跡ができます。

各プロセスのトラブルシュート資料

プロセスの特徴によって、それぞれ異なる資料に出力されるログやトレース情報です。プロセスの構成を次の図に示します。



プロセスの特徴に応じたトラブルシュート資料について次の表に示します。

項番	プロセス	プロセスの特徴と資料の用途	トラブルシュート資料
1	すべてのプロセス	OS 上で動作します。 OS から出力される情報をトラブルシュート資料として利用できます。	• OS のログ • OS の統計情報 • OS の状態情報 • メモリーダンプ（core ダンプ）
2	Java のプロセス • Java EE サーバ • ドメイン管理サーバ • asadmin	Java VM 上で動作します。 Java VM レイヤーについては、一般的な Java のトラブルシュート資料に加え、製品で拡張した Java ログやスレッドダンプがトラブルシュート資料として利用できます。	• Java VM ログ • スタックトレースログ • スレッドダンプ • エラーリポートファイル

項番	プロセス	プロセスの特徴と資料の用途	トラブルシュート資料
3	リクエストを処理するプロセス • Web サーバ • Java EE サーバ • HiRDB	複数のプロセスにわたってリクエスト処理が実行されます。 リクエスト処理の流れを複数プロセスにわたって追うことができます。	• Web サーバのアクセスログ • Web サーバのリクエストログ • 性能解析トレース
4	Java EE サーバ	セッションオブジェクトを明示管理ヒープで管理します。 明示管理ヒープ機能のイベントの発生がログで確認できます。	明示管理ヒープイベントログ
5		インプロセスで MQ Broker が動作します。 MQ Broker のメッセージがログで確認できます。	MQ Broker メッセージログ
6		JSP や JAX-WS の開発時に使用されるコマンドを提供します。 標準エラー出力でエラーメッセージが確認できます。	標準エラー出力
7	Java EE サーバ（asadmin）	Java EE サーバを起動します。 Java EE サーバのロガー初期化前のタイミングの障害情報が確認できます。	プロセス起動ログ

ドメイン管理サーバのトラブルシュート資料

ドメイン管理サーバのトラブルシュート資料の一覧を示します。

項番	ログの名称	出力先ディレクトリー	ファイル名	ラップ時間（初期値）	サイズ（初期値）	面数※1（初期値）	ログの設定方法
1	ドメイン管理サーバ メッセージログ※2	*HJES_LOGSDIR*/domains/*ドメイン名*	das_message*n*.log	00:00:00	16 メガバイト	8	asadmin ユーティリティーコマンドのset-log-attributesサブコマンドのパラメーターで指定します。
2	ドメイン管理サーバ スタックトレースログ※2	*HJES_LOGSDIR*/domains/*ドメイン名*	das_stacktrace*n*.log	00:00:00	16 メガバイト	8	asadmin ユーティリティーコマンドのset-log-attributesサブコマンドの

項番	ログの名称	出力先ディレクトリー	ファイル名	ラップ時間（初期値）	サイズ（初期値）	面数※1（初期値）	ログの設定方法
							パラメーターで指定します。
3	ドメイン管理サーバ Java VM ログ	*HJES_LOGSDIR*/domains/*ドメイン名*	das_javavm*n*.log	00:00:00	128 メガバイト	8	asadmin ユーティリティーコマンドの create-jvm-options サブコマンドのパラメーターで指定します。
4	ドメイン管理サーバ スレッドダンプ	*Application Serverのインストールディレクトリー*/javaee/glassfish/domains/*ドメイン名*/config	javacore*プロセスID*.*日時*.txt	• ユーザーの操作を契機に出力される。 • サイズ制限なし。 • 時間・面数による削除はなし。			なし
5	ドメイン管理サーバ エラーリポートファイル	*Application Serverのインストールディレクトリー*/javaee/glassfish/domains/*ドメイン名*/config	hs_err_pid*プロセスID*.log	• プロセスの異常終了を契機に出力される。 • サイズ制限なし。 • 時間・面数による削除はなし。			なし

注※1

バックアップ用のファイルに加えて書き込み用のファイルが 1 つ作成されます。例えば、面数が 8 の場合は合計 9 ファイルが作成されます。

注※2

スタックトレースの一部は、メッセージログに出力されます。

Java EE サーバ（サーバインスタンス）のトラブルシュート資料

Java EE サーバ（サーバインスタンス）のトラブルシュート資料の一覧を示します。

項番	ログの名称	出力先ディレクトリー	ファイル名	ラップ時間（初期値）	サイズ（初期値）	面数※1（初期値）	ログの設定方法
1	サーバインスタンス メッセージログ※2	*HJES_LOGSDIR*/nodes/*ノード名*/*サーバインスタンス名*	je_message*n*.log	00:00:00	16 メガバイト	8	asadmin ユーティリティーコマンドの set-log-attributes サブコマンドのパラメーターで指定します。

項番	ログの名称	出力先ディレクトリー	ファイル名	ラップ時間（初期値）	サイズ（初期値）	面数※1（初期値）	ログの設定方法
2	サーバインスタンス スタックトレースログ※2		je_stacktrace*n*.log	00:00:00	16 メガバイト	8	asadmin ユーティリティーコマンドのset-log-attributes サブコマンドのパラメーターで指定します。
3	WebSocket アクセスログ		je_websocket_access*n*.log	00:00:00	300 メガバイト	8	asadmin ユーティリティーコマンドのset-log-attributes サブコマンドのパラメーターで指定します。
4	サーバインスタンス Java VM ログ		je_javavm*n*.log	00:00:00	128 メガバイト	8	asadmin ユーティリティーコマンドの create-jvm-options サブコマンドのパラメーターで指定します。
5	サーバインスタンス 明示管理ヒープイベントログ		je_eheap_event*n*.log	00:00:00	128 メガバイト	8	asadmin ユーティリティーコマンドの create-jvm-options サブコマンドのパラメーターで指定します。
6	サーバインスタンス スレッドダンプログ	*Application Serverのインストールディレクトリー*/javaee/glassfish/nodes/*ノード名*/*サーバインスタンス名*/config	javacore*プロセスID*.*日時*.txt	• ユーザーの操作を契機に出力される。 • サイズ制限なし。 • 時間・面数による削除はなし。			なし
7	サーバインスタンス エラーリポートファイル		hs_err_pid*プロセスID*.log	• プロセスの異常終了を契機に出力される。 • サイズ制限なし。 • 時間・面数による削除はなし。			なし

注※1

バックアップ用のファイルに加えて書き込み用のファイルが 1 つ作成されます。例えば、面数が 8 の場合は合計 9 ファイルが作成されます。

注※2

スタックトレースの一部は、メッセージログに出力されます。ユーザーが作成したアプリケーションによって出力される標準出力および標準エラー出力も、メッセージログに出力されます。

Java EE サーバ（asadmin）のトラブルシュート資料

Java EE サーバ（asadmin）のトラブルシュート資料の一覧を示します。

項番	ログの名称	出力先ディレクトリー	ファイル名	ラップ時間（初期値）	サイズ（初期値）	面数※1（初期値）	ログの設定方法
1	asadmin メッセージログ※2	*HJES_LOGSDIR*/commands/asadmin	asadmin_messag*e*n.log	00:00:00	16 メガバイト	8	asenv の環境変数に指定します。
2	asadmin スタックトレースログ※2		asadmin_stackt*race*n.log	00:00:00	16 メガバイト	8	asenv の環境変数に指定します。
3	asadmin スレッドダンプ	*コマンド実行時のカレントディレクトリー*	javacore*プロセスID*.*日時*.txt	• ユーザーの操作を契機に出力される。 • サイズ制限なし。 • 時間・面数による削除はなし。			なし
4	asadmin エラーリポートファイル		hs_err_pid*プロセスID*.log	• プロセスの異常終了を契機に出力される。 • サイズ制限なし。 • 時間・面数による削除はなし。			なし
5	asadmin プロセス起動時ログ	*HJES_LOGSDIR*/commands/asadmin	asadmin_launch*n*.log	00:00:00	16 メガバイト	8	asenv の環境変数に指定します。

注※1

バックアップ用のファイルに加えて書き込み用のファイルが 1 つ作成されます。例えば、面数が 8 の場合は合計 9 ファイルが作成されます。

注※2

スタックトレースの一部は、メッセージログに出力されます。

Java EE サーバ（アプリケーションクライアント）のトラブルシュート資料

Java EE サーバ（アプリケーションクライアント）のトラブルシュート資料の一覧を示します。

項番	ログの名称	出力先ディレクトリー	ファイル名	ラップ時間（初期値）	サイズ（初期値）	面数※（初期値）	ログの設定方法
1	アプリケーションクライアントスレッドダンプログ	*カレントディレクトリー*（appclientコマンドを起動し	javacore*プロセスID*.*日時*.txt	• ユーザーの操作を契機に出力される。 • サイズ制限なし。 • 時間・面数による削除はなし。			なし

項番	ログの名称	出力先ディレクトリー	ファイル名	ラップ時間（初期値）	サイズ（初期値）	面数※（初期値）	ログの設定方法
2	アプリケーションクライアントエラーリポートファイル	たディレクトリー）	hs_err_pid*プロセスID*.log	• プロセスの異常終了を契機に出力される。 • サイズ制限なし。 • 時間・面数による削除はなし。			なし

注※

バックアップ用のファイルに加えて書き込み用のファイルが 1 つ作成されます。例えば、面数が 8 の場合は合計 9 ファイルが作成されます。

Java EE サーバ（MQ Broker） のトラブルシュート資料

Java EE サーバ（MQ Broker）のトラブルシュート資料の一覧を示します。

項番	ログの名称	出力先ディレクトリー	ファイル名	ラップ時間（初期値）	サイズ（初期値）	面数※1（初期値）	ログの設定方法
1	MQ Broker メッセージログ	*Application Serverのインストールディレクトリー*/javaee/nodes/*ノード名*/*サーバインスタンス名*/imq/instances/*MQインスタンス名*/log	log_*n*※2.txt	00:00:00	16 メガバイト	8	config.properties ファイルに指定します。

注※1

バックアップ用のファイルに加えて書き込み用のファイルが 1 つ作成されます。例えば、面数が 8 の場合は合計 9 ファイルが作成されます。

注※2

n ＝ 0～面数-1

Web サーバのトラブルシュート資料

Web サーバのトラブルシュート資料の一覧を示します。

項番	ログの名称	出力先ディレクトリー	ファイル名	ラップ時間（初期値）	サイズ（初期値）	面数（初期値）	ログの設定方法
1	アクセスログ	*HJES_LOGSDIR*/nodes/*ノード名*/*Webサーバ名*	access.*n*※1	24 時間	最大 2 ギガバイト	8	TransferLog ディレクティブまたは CustomLog ディ

項番	ログの名称	出力先ディレクトリー	ファイル名	ラップ時間（初期値）	サイズ（初期値）	面数（初期値）	ログの設定方法
							レクティブで指定します。※2
2	リクエストログ		hwsrequest.*n*※1	24 時間	最大 2 ギガバイト	8	HWSRequestLog ディレクティブで指定します。※2
3	エラーログ		error.n※1	24 時間	最大 2 ギガバイト	8	ErrorLog ディレクティブで指定します。※2
4	WebSocket リクエストログ		hws_websocket_log.*n*※3	-（ログファイルサイズでローテート）	512 メガバイト	2	HWSWebSocketLog ディレクティブで指定します。※2
5	プロセス ID ログ		httpd.pid	-	-	-	PidFile ディレクティブで指定します。※2
6	内部トレース		hws.trclog.*n*※4	-	-	5	HWSTraceLogFile ディレクティブで指定します。※2
7	共有メモリー ID ログ		hws.trcid	-	-	-	HWSTraceIdFile ディレクティブで指定します。※5

（凡例）

-：該当なし

注※1

n ＝ログ採取開始時刻

注※2

出力先ディレクトリー、ファイル名を変更した場合、システム情報収集機能で収集できません。手動で収集する必要があります。

注※3

n=001～256

製品の初期設定の rotatelogs2 を使用した場合の値です。

注※4

n=01～05

注※5

システム情報収集機能で収集できません。手動で収集する必要があります。

パフォーマンストレーサーのトラブルシュート資料（トラブルシュート用）

パフォーマンストレーサーのトラブルシュート資料（トラブルシュート用）の一覧を示します。

項番	ログの名称	出力先ディレクトリー	ファイル名	ラップ時間（初期値）	サイズ（初期値）	面数（初期値）	ログの設定方法
1	メッセージログ	*HJES_LOGSDIR*/nodes/*ノード名*/*PRF識別子*/log/*PRF識別子*	prf_message*n*※	00:00:00	10 メガバイト	8	なし

注※

n=01～32

パフォーマンストレーサーのトラブルシュート資料（性能解析用）

パフォーマンストレーサーのトラブルシュート資料（性能解析用）の一覧を示します。

項番	ログの名称	出力先ディレクトリー	ファイル名	ラップ時間（初期値）	サイズ（初期値）	面数（初期値）	ログの設定方法
1	PRF トレース	*HJES_LOGSDIR*/nodes/*ノード名*/*PRF識別子*/utt/prf/*PRF識別子*/dcopltrc	prf_*n*※1	時間でラップしません。	8 メガバイト	4	なし
2	性能解析トレースファイル（PRF トレースを CSV 形式に編集したファイル）	システム情報収集機能によって収集されたアーカイブファイル内。 （例）アーカイブファイルの出力先がD:/snapshot の場合は、性能解析トレースファイルは次のフォルダに zip 形式で格納されます。 *アーカイブファイルのルート*/D/snapshot/	*node名-サーバインスタンス名-ファイル生成日時*.csv※2	• システム情報収集機能の実行を契機に PRF トレースが CSV 形式に変換される。 • サイズ制限なし。 • 時間・面数による削除はなし。			なし

項番	ログの名称	出力先ディレクトリー	ファイル名	ラップ時間（初期値）	サイズ（初期値）	面数（初期値）	ログの設定方法
		作業用ディレクトリ/prfTrace/					

注※1

n=001～256

注※2

zip ファイルを展開したファイルです。

9.2　ログファイルの出力形式について

各プロセスから出力されるログファイルのローテーション方式と出力形式について説明します。

9.2.1　ログファイルのローテーション方式

各プロセスのログは、複数のファイルにローテーションして出力されます。指定した時刻または出力先のファイルサイズが一定のサイズになったタイミングで、出力先ファイルを切り替えて出力されます。各プロセスのログのローテーション方式について説明します。

Java EE サーバのログのローテーション方式

ログファイルがprf_message01 だった場合、ローテーションが発生すると、ログファイルをprf_message02 にリネームし、新しく作成したprf_message01 にログを出力します。2 回目のローテーションが発生すると、prf_message01 をprf_message02 に、prf_message02 をprf_message03 にリネームし、ログをprf_message01 に出力します。 指定したファイル数を超えると、番号が最も大きいファイルを削除します。

Web サーバのログのローテーション方式

rotatelogs プログラムまたは rotatelogs2 プログラムを使用してログファイルの分割方法を指定できます。分割方法は、Web サーバの定義ファイルに指定します。

rotatelogs プログラム

アクセスログやエラーログを一定時間単位（例えば、24 時間ごと）に分割して、複数のファイルに出力できます。

- ログ分割時間間隔

  1 つのログファイルを採取する時間間隔を指定します。指定した時間が経過するごとに、新規ファイルにログを採取します。

- ファイル数

  分割したログファイルのファイル数を指定します。指定したファイル数を超えた場合、最も古いファイルから削除されます。

rotatelogs2 プログラム

アクセスログやエラーログをログファイルサイズで分割して、複数のファイルにラップアラウンドして出力できます。

- ログファイルサイズ

  ログファイルの最大サイズを指定します（単位：キロバイト）。ログを出力するタイミングで、最大サイズを超えていると、次のログファイルをクリアして続きが出力されます。

- ログファイル個数

出力するログファイルの最大個数を指定します。最大サイズを超えて次のログファイルに移る場合、それまで処理していたログファイルの拡張子が最大個数と同じとき、再度 1 つ目のファイルから使用します。

(例) ログファイル個数を3 に指定した場合

ログファイルがerrorlog.001 の場合、errorlog.001 からerrorlog.003 の順番にログが出力されます。errorlog.003 が指定したログファイルサイズを超えると、errorlog.001 をクリアして続きが出力されます。

パフォーマンストレーサーのログのローテーション方式

prf_message01 が常に最新のログファイルです。ローテーションが発生すると、prf_message01 を prf_message02 にリネームします。2 回目のローテーション事案が発生すると、prf_message01 を prf_message03 にリネームします(prf_message02 からprf_message03 へのリネームは行わない)。

アプリケーション開発環境のログのローテーション方式

指定したファイルサイズを契機にログファイルをリネームして、出力します。

9.2.2 Java EE サーバのログの出力形式

Java EE サーバが提供するログの出力形式と出力項目について説明します。

Java EE サーバの提供するログ

Java EE サーバが提供するログの種別を次に示します。

- メッセージログ
- スタックトレースログ
- Java VM ログ
- 明示管理ヒープ機能のイベントログ
- スレッドダンプログ
- プロセス起動時ログ
- WebSocket アクセスログ
- 保守ログ

それぞれのログの出力形式と、出力項目について説明します。

メッセージログ

出力形式

```
番号 日付 時刻 アプリケーション名 pid tid メッセージID  テキスト
```

出力項目

出力項目	説明
番号	トレースレコードの通番（4 けた）。
日付	トレースの取得日付。*yyyy/mm/dd* 形式で出力されます。
時刻	トレースの取得時刻。*hh:mm:ss.sss* 形式で出力されます。
アプリケーション名	アプリケーション識別名。 アプリケーションを特定するための名称です。
pid	プロセス ID。
tid	スレッド識別子。 スレッドを識別するための ID です。
メッセージ ID	メッセージ ID。 メッセージを区別するための ID です。
テキスト	メッセージテキスト。

ログレベル

ログレベル	概要
1	システムが続行できなくなる障害に関するメッセージや運用時に必須となるメッセージなどを出力します。
2	レベル 1 のメッセージに加え、障害になるおそれがある事象に関する警告メッセージなどを出力します。
3	レベル 2 のメッセージに加え、稼働情報に関するメッセージなどを出力します。 レベル 3 は開発環境だけで使用し、実行環境では使用しないでください。

スタックトレースログ

出力形式

メッセージログと同じです。

出力項目

メッセージログと同じです。

Java VM ログ

拡張 verbosegc 出力機能(G1GC 使用時以外)

出力形式

```
[id] date (Skip Full:full_count, Copy:copy_count)
[gc_kind gc_info, gc_time secs][Eden: eden_info][Survivor: survivor_info]
[Tenured: tenured_info][Metaspace: metaspace_info]
[class space: class_space_info] [cause:cause_info] [User: user_cpu secs]
[Sys: system_cpu secs][IM: jvm_alloc_size, mmap_total_size, malloc_total_size]
```

```
[TC: thread_count]
[DOE: doe_alloc_size, called_count][CCI: cc_used_sizeK, cc_max_sizeK, cc_infoK]
```

注

改行はありません。

出力項目

出力項目	説明
id	Java VM ログファイル識別子。
date	GC の開始日時。-XX:-HitachiVerboseGCPrintDate オプション指定時は出力されません。
full_count	FullGC をスキップした回数。-XX:HitachiVerboseGCIntervalTime オプション指定時に出力されます。
copy_count	CopyGC をスキップした回数。-XX:HitachiVerboseGCIntervalTime オプション指定時に出力されます。
gc_kind	GC 種別（Full GC またはGC）。
gc_info	GC 情報（GC 前の領域長 -> GC 後の領域長(領域サイズ)） (例) 264K->0K(512K)。
gc_time	GC 経過時間（単位：秒）。
Eden	Eden 領域の種別（DefNew::Eden）。
eden_info	Eden 領域のメモリー情報。
Survivor	Survivor 領域の種別（DefNew::Survivor）。
survivor_info	Survivor 領域のメモリー情報。
Tenured	Tenured 領域の種別（Tenured）。
tenured_info	Tenured 領域のメモリー情報。
metaspace_info	Metaspace 領域のメモリー情報（単位：キロバイト）。
classspace_info	CompressedClassSpace 情報（単位：キロバイト）。
cause_info	GC 要因内容。
user_cpu	GC スレッドがユーザーモードに費やした CPU 時間（単位：秒）。
system_cpu	GC スレッドがカーネルモードに費やした CPU 時間（単位：秒）。
jvm_alloc_size	Java VM 内部で管理している領域のうち、現在使用中の領域のサイズ。
mmap_total_size	Java VM 内部で管理している領域のうち、mmap で割り当てた C ヒープ領域の総サイズ。-XX:+HitachiVerboseGCPrintJVMInternalMemory オプション指定時に出力されます。
malloc_total_size	Java VM 内部で管理している領域のうち、malloc で割り当てた C ヒープの総 サイズ。-XX:+HitachiVerboseGCPrintJVMInternalMemory オプション指定時に出力されます。
thread_count	Java スレッドの数。-XX:+HitachiVerboseGCPrintThreadCount オプション指定時に出力されます。

出力項目	説明
doe_alloc_size	java.io.File.deleteOnExit()を呼び出して確保した累積のヒープサイズ。-XX:+HitachiVerboseGCPrintDeleteOnExit オプション指定時に出力されます。
called_count	java.io.File.deleteOnExit()の呼び出し回数。-XX:+HitachiVerboseGCPrintDeleteOnExit オプション指定時に出力されます。
cc_used_size	GC 時のコードキャッシュ領域の使用サイズ（単位：キロバイト）。-XX:+PrintCodeCacheInfo オプション指定時に出力されます。
cc_max_size	コードキャッシュ領域の最大サイズ（単位：キロバイト）。-XX:+PrintCodeCacheInfo オプション指定時に出力されます。
cc_info	保守情報。-XX:+PrintCodeCacheInfo オプション指定時に出力されます。

拡張 verbosegc 出力機能(G1GC 使用時)

出力形式

```
[id]date[gc_kind gc_info, gc_time secs][Status:gc_status]
[G1GC::Eden: eden_info][G1GC::Survivor: survivor_info]
[G1GC::Tenured: tenured_info][G1GC::Humongous: humongous_info]
[G1GC::Free: free_info][Metaspace: metaspace_info]
[class space: class_space_info] [cause:cause_info][RegionSize: region_sizeK]
[Target: target_time secs][Predicted: predicted_time secs]
[TargetTenured: target_sizeK][Reclaimable: reclaimable_info]
[User: user_cpu secs][Sys: system_cpu secs]
[IM: jvm_alloc_sizeK, mmap_total_sizeK, malloc_total_sizeK]
[TC: thread_count][DOE: doe_alloc_sizeK, called_count]
[CCI: cc_used_sizeK, cc_max_sizeK, cc_infoK]
```

注

改行はありません。

出力項目

出力項目	説明
id	Java VM ログファイル識別子。
date	GC または CM の開始日時。
gc_kind	GC または CM の種別。 Full GC、Mixed GC、Young GC、Young GC(initial-mark)、CM Remark、または、CM Cleanup が出力されます。
gc_info	Java ヒープ領域のメモリー情報（単位：キロバイト）。
gc_time	GC 経過時間（単位：秒）。
gc_status	GC の状態（to exhausted または-）。 gc_kind がYoung GC、Young GC(initial-mark)または、Mixed GC のどれかの場合 -またはto exhausted が出力されます。 gc_kind がYoung GC、Young GC(initial-mark)または、Mixed GC 以外の場合 -が出力されます。

出力項目	説明
eden_info	Eden 領域のメモリー情報（単位：キロバイト）。
survivor_info	Survivor 領域のメモリー情報（単位：キロバイト）。
tenured_info	Tenured 領域のメモリー情報（単位：キロバイト）。
humongous_info	Humongous 領域のメモリー情報（単位：キロバイト）。
free_info	Free 領域のメモリー情報（単位：キロバイト）。
metaspace_info	Metaspace 領域のメモリー情報（単位：キロバイト）。
classspace_info	CompressedClassSpace 情報（単位：キロバイト）。
cause_info	GC 要因内容。
region_size	1 リージョンのサイズ（単位：キロバイト）。
target_time	GC によるアプリケーション停止時間の目標時間（単位：秒）。
predicted_time	Java VM が予測した GC によるアプリケーション停止時間（単位：秒）。
target_size	Mixed GC で GC 対象となった Tenured 領域のサイズ。
reclaimable_info	予測回収サイズ情報（単位：キロバイト）。
user_cpu	GC スレッドがユーザーモードに費やした CPU 時間（単位：秒）。-XX:-HitachiVerboseGCCpuTime オプション指定時は出力されません。 CPU 時間取得に失敗した場合にはunknown が表示されます。
system_cpu	GC スレッドがカーネルモードに費やした CPU 時間（単位：秒）。 -XX:-HitachiVerboseGCCpuTime オプション指定時は出力されません。 CPU 時間取得に失敗した場合にはunknown が表示されます。

CSV 形式出力(G1GC 使用時以外)

出力形式

```
id,date,full_count,copy_count,gc_kind,gc_info,gc_time,eden_info,
survivor_info,tenured_info,metaspace_info,classspace_info,cause_info,user_cpu,
system_cpu,jvm_alloc_size,mmap_total_size,malloc_total_size,thread_count,
doe_alloc_size,called_count,cc_used_size,cc_max_size,cc_info
```

注

改行はありません。

出力項目

出力項目	説明
id	Java VM ログファイル識別子。
date	GC の開始日時。
full_count	FullGC をスキップした回数（-XX:HitachiVerboseGCIntervalTime オプション指定時に出力されます）。

出力項目	説明
copy_count	CopyGC をスキップした回数。-XX:HitachiVerboseGCIntervalTime オプション指定時に出力されます。
gc_kind	GC 種別（Full GC またはGC）。
gc_info	GC 情報（GC 前の領域長 -> GC 後の領域長(領域サイズ)）（単位：キロバイト）。 (例) 264K->0K(512K)。
gc_time	GC 経過時間（単位：秒）。
eden_info	Eden 領域情報（単位：キロバイト）。
survivor_info	Survivor 領域のメモリー情報（単位：キロバイト）。
tenured_info	Tenured 領域のメモリー情報（単位：キロバイト）。
metaspace_info	Metaspace 領域のメモリー情報（単位：キロバイト）。
classspace_info	CompressedClassSpace 情報（単位：キロバイト）。 圧縮オブジェクトポインタ機能が無効の場合はダミーで 0 を出力します。
cause_info	GC 要因番号。-XX:-HitachiVerboseGCPrintCause オプション指定時は出力さません。
user_cpu	GC スレッドがユーザーモードに費やした CPU 時間（単位：秒）。 CPU 時間取得に失敗した場合にはunknown が表示されます。-XX:-HitachiVerboseGCCpuTime オプション指定時は出力されません。
system_cpu	GC スレッドがカーネルモードに費やした CPU 時間（単位：秒）。 CPU 時間取得に失敗した場合にはunknown が表示されます。-XX:-HitachiVerboseGCCpuTime オプション指定時は出力されません。
jvm_alloc_size	Java VM 内部で管理している領域のうち、現在使用中の領域のサイズ。
mmap_total_size	Java VM 内部で管理している領域のうち、mmap で割り当てた C ヒープ領域の総サイズ。-XX:-HitachiVerboseGCPrintJVMInternalMemory オプション指定時は出力されません。
malloc_total_size	Java VM 内部で管理している領域のうち、malloc で割り当てた C ヒープ領域の総サイズ。-XX:-HitachiVerboseGCPrintJVMInternalMemory オプション指定時は出力されません。
thread_count	Java スレッドの数。-XX:-HitachiVerboseGCPrintJVMInternalMemory オプション指定時は出力されません。
doe_alloc_size	java.io.File.deleteOnExit()を呼び出して確保した累積のヒープサイズ-XX:-HitachiVerboseGCPrintDeleteOnExit オプション指定時は出力されません。
called_count	java.io.File.deleteOnExit()の呼び出し回数。-XX:-HitachiVerboseGCPrintDeleteOnExit オプション指定時は出力されません。
cc_used_size	GC 時のコードキャッシュ領域の使用サイズ（単位：キロバイト）。-XX:-PrintCodeCacheInfo オプション指定時は出力されません。
cc_max_size	コードキャッシュ領域の最大サイズ（単位：キロバイト）。-XX:-PrintCodeCacheInfo オプション指定時は出力されません。
cc_info	保守情報。-XX:-PrintCodeCacheInfo オプション指定時は出力されません。

CSV 形式出力(G1GC 使用時)

出力形式

```
id,date,gc_kind,gc_info,gc_time,gc_status,eden_info,survivor_info,
tenured_info,humongous_info,free_info,metaspace_info,
classspace_info,cause_info,region_size,target_time,predicted_time,
target_size,reclaimable_info,user_cpu,system_cpu,jvm_alloc_size,
mmap_total_size,malloc_total_size,thread_count,doe_alloc_size,
called_count,cc_used_size,cc_max_size,cc_info
```

注

改行はありません。

出力項目

出力項目	説明
id	Java VM ログファイル識別子。
date	GC の開始日時。
gc_kind	GC または CM の種別。 Full GC、Mixed GC、Young GC、 Young GC(initial-mark)、CM Remark、または、 CM Cleanup が出力されます。
gc_info	Java ヒープ領域のメモリー情報（単位：キロバイト)。
gc_time	GC によるアプリケーション停止時間（単位：秒)。
gc_status	GC の状態（to exhausted または-)。 gc_kind がYoung GC、Young GC(initial-mark)または、Mixed GC のどれかの場合 -またはto exhausted が出力されます。 gc_kind がYoung GC、Young GC(initial-mark)または、Mixed GC 以外の場合 -が出力されます。
eden_info	Eden 領域のメモリー情報（単位：キロバイト)。
survivor_info	Survivor 領域のメモリー情報（単位：キロバイト)。
tenured_info	Tenured 領域のメモリー情報（単位：キロバイト)。
humongous_info	Humongous 領域のメモリー情報（単位：キロバイト)。
free_info	Free 領域のメモリー情報（単位：キロバイト)。
metaspace_info	Metaspace 領域のメモリー情報（単位：キロバイト)。
classspace_info	CompressedClassSpace 情報（単位：キロバイト)。
cause_info	GC 要因内容。
region_size	1 リージョンのサイズ（単位：キロバイト)。
target_time	GC によるアプリケーション停止時間の目標時間（単位：秒)。
predicted_time	Java VM が予測した GC によるアプリケーション停止時間（単位：秒)。
target_size	Mixed GC で GC 対象となった Tenured 領域のサイズ。

出力項目	説明
reclaimable_info	予測回収サイズ情報（単位：キロバイト）。
user_cpu	全 GC スレッドがユーザーモードに費やした CPU 時間（単位：秒）。-XX:-HitachiVerboseGCCpuTime オプション指定時は出力されません。 CPU 時間取得に失敗した場合にはunknown が表示されます。
system_cpu	全スレッドがカーネルモードに費やした CPU 時間（単位：秒）。 -XX:-HitachiVerboseGCCpuTime オプション指定時は出力されません。 CPU 時間取得に失敗した場合にはunknown が表示されます。

明示管理ヒープ機能のイベントログ

明示管理ヒープ機能のイベントログの出力契機は、設定しているログ出力レベルに応じて、イベントが異なります。

ログ出力レベルには、normal、verbose、debug の 3 種類があります。各レベルごとの出力形式を示します。

normal の場合（GC 時の Explicit ヒープの利用状況）

出力形式

```
[ENS]ctime[EH: EH_USED_BF->EH_USED_AF(EH_TOTAL/EH_MAX)]
[E/F/D: AC_NUM/FL_NUM/DA_NUM][cause:CAUSE][CF: CF_CNT]
```

注

改行はありません。

出力項目

出力項目	説明
ctime	GC 発生の日時を示します。拡張 verbosegc 機能で出力しているものと同一の時刻形式で出力されます。 HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
EH_USED_BF	GC 前の、Explicit ヒープの利用済みサイズが出力されます（単位：キロバイト）。
EH_USED_AF	GC 後の、Explicit ヒープ利用済みサイズ。
EH_TOTAL	GC 後の、確保済み Explicit ヒープサイズ（単位：キロバイト）。
EH_MAX	Explicit ヒープ最大サイズ（単位：キロバイト）。
AC_NUM	GC 後の、サブ状態がEnable である有効な Explicit メモリーブロック数。
FL_NUM	拡張用の出力項目であり、常に 0 が出力されます。
DA_NUM	GC 後の、サブ状態がDisable である有効な Explicit メモリーブロック数。
CAUSE	GC は Copy GC、Full GC は Full GC が契機となって、このログを出力したことを示します。
CF_CNT	前回の GC から今回の GC までの間に、Explicit メモリーブロックの初期化に失敗した回数。

normal の場合（Explicit メモリーブロック解放処理）

出力形式

```
[ENS]ctime[EH: EH_USED_BF->EH_USED_AF(EH_TOTAL/EH_MAX),
ELAPSED secs][E/F/D: AC_NUM/FL_NUM/DA_NUM]
[DefNew::Eden: ED_USED_BF->ED_USED_AF(ED_TOTAL)]
[DefNew::Survivor: SV_USED_BF->SV_USED_AF(SV_TOTAL)]
[Tenured: TN_USED_BF->TN_USED_AF(TN_TOTAL)][User: USERCPU secs]
[Sys: SYSCPU secs][cause:CAUSE]
```

出力項目

出力項目	説明
ctime	Explicit メモリーブロック解放処理発生の日時。拡張 verbosegc 機能で出力されるログと同一の時刻形式。HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
EH_USED_BF	Explicit メモリーブロック解放処理前の、Explicit ヒープ利用済みサイズ（単位：キロバイト）。
EH_USED_AF	Explicit メモリーブロック解放処理後の、Explicit ヒープ利用済みサイズ（単位：キロバイト）。
EH_TOTAL	Explicit メモリーブロック解放処理後の、確保済み Explicit ヒープサイズ（単位：キロバイト）。
EH_MAX	Explicit ヒープ最大サイズ（単位：キロバイト）。
ELAPSED	Explicit メモリーブロック解放処理に要した時間（単位：秒）。
AC_NUM	Explicit メモリーブロック解放処理後の、サブ状態がEnable である有効な Explicit メモリーブロック数。
FL_NUM	拡張用の出力項目であり、常に 0 を示します。
DA_NUM	Explicit メモリーブロック解放処理後の、サブ状態がDisable である有効な Explicit メモリーブロック数。
ED_USED_BF	Explicit メモリーブロック解放処理前の、Eden 領域利用済みサイズ（単位：キロバイト）。
ED_USED_AF	Explicit メモリーブロック解放処理後の、Eden 領域利用済みサイズ（単位：キロバイト）。
ED_TOTAL	Explicit メモリーブロック解放処理後の、Eden 領域確保済みサイズ（単位：キロバイト）。
SV_USED_BF	Explicit メモリーブロック解放処理前の、Survivor 領域利用済みサイズ（単位：キロバイト）。
SV_USED_AF	Explicit メモリーブロック解放処理後の、Survivor 領域利用済みサイズ（単位：キロバイト）。
SV_TOTAL	Explicit メモリーブロック解放処理後の、Survivor 領域確保済みサイズ（単位：キロバイト）。
TN_USED_BF	Explicit メモリーブロック解放処理前の、Tenured 領域利用済みサイズ（単位：キロバイト）。
TN_USED_AF	Explicit メモリーブロック解放処理後の、Tenured 領域利用済みサイズ（単位：キロバイト）。
TN_TOTAL	Explicit メモリーブロック解放処理後の、Tenured 領域確保済みサイズ（単位：キロバイト）。

出力項目	説明
USERCPU	Explicit メモリーブロック解放処理に要したユーザー CPU 時間（単位：秒）。
SYSCPU	Explicit メモリーブロック解放処理に要したシステム CPU 時間（単位：秒）。
CAUSE	必ずReclaim と出力します。Explicit メモリーブロックの解放で出力したログであることを示します。

normal の場合（Explicit メモリーブロック解放処理時の Java ヒープあふれ）

出力形式

```
[ENS]ctime[EH: EH_USED_BF->EH_USED_AF(EM_TOTAL/EH_MAX),
ELAPSED secs][E/F/D: AC_NUM/FL_NUM/DA_NUM]
[DefNew::Eden: ED_USED_BF->ED_USED_AF(ED_TOTAL)]
[DefNew::Survivor: SV_USED_BF->SV_USED_AF(SV_TOTAL)]
[Tenured: TN_USED_BF->TN_USED_AF(TN_TOTAL)][User: USERCPU secs]
[Sys: SYSCPU secs][cause:CAUSE]
```

出力項目

出力項目	説明
ctime	Explicit メモリーブロック解放処理発生の日時。拡張 verbosegc 機能で出力されるログと同一の時刻形式。HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
EH_USED_BF	Explicit メモリーブロック解放処理前の、Explicit ヒープ利用済みサイズ（単位：キロバイト）。
EH_USED_AF	Java ヒープがあふれたあとの、Explicit ヒープ利用済みサイズ。Java ヒープがあふれた場合は、Explicit メモリーブロックの解放処理を行なわないため、必ず *EH_USED_BF* と同じ値になります（単位：キロバイト）。
EH_TOTAL	Java ヒープがあふれたあとの、確保済み Explicit ヒープサイズ（単位：キロバイト）。
EH_MAX	Explicit ヒープ最大サイズ（単位：キロバイト）。
ELAPSED	Explicit メモリーブロック解放処理開始から、Java ヒープがあふれるまでの時間（単位：秒）。
AC_NUM	Java ヒープがあふれたあとの、サブ状態がEnable である有効な Explicit メモリーブロック数。
FL_NUM	拡張用の出力項目であり、常に 0 を示します。
DA_NUM	Java ヒープがあふれたあとの、サブ状態がDisable である有効な Explicit メモリーブロック数。
ED_USED_BF	Explicit メモリーブロック解放処理前の、Eden 領域利用済みサイズ（単位：キロバイト）。
ED_USED_AF	Java ヒープがあふれたあとの、Eden 領域利用済みサイズ（単位：キロバイト）。
ED_TOTAL	Java ヒープがあふれたあとの、Eden 領域確保済みサイズ（単位：キロバイト）。
SV_USED_BF	Explicit メモリーブロック解放処理前の、Survivor 領域利用済みサイズ（単位：キロバイト）。
SV_USED_AF	Java ヒープがあふれたあとの、Survivor 領域利用済みサイズ（単位：キロバイト）。
SV_TOTAL	Java ヒープがあふれたあとの、Survivor 領域確保済みサイズ（単位：キロバイト）。

出力項目	説明
TN_USED_BF	Explicit メモリーブロック解放処理前の、Tenured 領域利用済みサイズ（単位：キロバイト）。
TN_USED_AF	Java ヒープがあふれたあとの、Tenured 領域利用済みサイズ（単位：キロバイト）。
TN_TOTAL	Java ヒープがあふれたあとの、Tenured 領域確保済みサイズ（単位：キロバイト）。
USERCPU	Explicit メモリーブロック解放処理開始から、Java ヒープがあふれるまでのユーザー CPU 時間（単位：秒）。
SYSCPU	Explicit メモリーブロック解放処理開始から、Java ヒープがあふれるまでのシステム CPU 時間（単位：秒）。
CAUSE	必ずReclaiming と出力されます。Explicit メモリーブロック解放処理時の Java ヒープあふれで出力したログであることを示します。

normal の場合（Explicit メモリーブロックのマイグレート処理）

出力形式

```
[ENS]ctime[EH: EH_USED_BF->EH_USED_AF(EH_TOTAL>/EH_MAX),
ELAPSED secs][E/F/D: AC_NUM/FL_NUM/DA_NUM]
[DefNew::Eden: ED_USED_BF->ED_USED_AF>(ED_TOTAL)]
[DefNew::Survivor: SV_USED_BF->SV_USED_AF(SV_TOTAL)]
[Tenured: TN_USED_BF->TN_USED_AF(TN_TOTAL)]
[target:EH_MIG_TRG/EH_MIG_DED/EH_MIG_LIV]
[User: USERCPU secs][Sys: SYSCPU secs][cause:CAUSE]
```

出力項目

出力項目	説明
ctime	Explicit メモリーブロックの自動マイグレート予約発生の日時。拡張 verbosegc 機能で出力しているものと同一の時刻形式。 HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
EH_USED_BF	Explicit メモリーブロックのマイグレート処理前の、Explicit ヒープ利用済みサイズ（単位：キロバイト）。
EH_USED_AF	Explicit メモリーブロックのマイグレート処理後の、Explicit ヒープ利用済みサイズ（単位：キロバイト）。
EH_TOTAL	Explicit メモリーブロックのマイグレート処理後の、確保済み Explicit ヒープサイズ（単位：キロバイト）
EH_MAX	Explicit ヒープ最大サイズ（単位：キロバイト）。
ELAPSED	Explicit メモリーブロックの自動マイグレート予約の処理の開始から、マイグレート処理終了までの時間（単位：秒）。
AC_NUM	Explicit メモリーブロックのマイグレート処理後の、サブ状態がEnable である有効な Explicit メモリーブロック数。
FL_NUM	拡張用の出力項目であり、常に 0 を示します。
DA_NUM	Explicit メモリーブロックのマイグレート処理後の、サブ状態がDisable である有効な Explicit メモリーブロック数。

出力項目	説明
ED_USED_BF	Explicit メモリーブロックのマイグレート処理前の、Eden 領域利用済みサイズ（単位：キロバイト）。
ED_USED_AF	Explicit メモリーブロックのマイグレート処理後の、Eden 領域利用済みサイズ（単位：キロバイト）。
ED_TOTAL	Explicit メモリーブロックのマイグレート処理後の、Eden 領域確保済みサイズ（単位：キロバイト）。
SV_USED_BF	Explicit メモリーブロックのマイグレート処理前の、Survivor 領域利用済みサイズ（単位：キロバイト）。
SV_USED_AF	Explicit メモリーブロックのマイグレート処理後の、Survivor 領域利用済みサイズ（単位：キロバイト）。
SV_TOTAL	Explicit メモリーブロックのマイグレート処理後の、Survivor 領域確保済みサイズ（単位：キロバイト）。
TN_USED_BF	Explicit メモリーブロックのマイグレート処理前の、Tenured 領域利用済みサイズ（単位：キロバイト）。
TN_USED_AF	Explicit メモリーブロックのマイグレート処理後の、Tenured 領域利用済みサイズ（単位：キロバイト）。
TN_TOTAL	Explicit メモリーブロックのマイグレート処理後の、Tenured 領域確保済みサイズ（単位：キロバイト）。
EH_MIG_TRG	Explicit メモリーブロックのマイグレート処理を行った Explicit ヒープの利用済みサイズ（単位：キロバイト）。
EH_MIG_DED	Explicit メモリーブロックのマイグレート処理を行ったことによって減少した、Explicit ヒープの利用済みサイズ（単位：キロバイト）。
EH_MIG_LIV	Explicit メモリーブロックのマイグレート処理を行ったことによって減少しなかった、Explicit ヒープの利用済みサイズ（単位：キロバイト）。
USERCPU	Explicit メモリーブロックの自動マイグレート予約の処理の開始から、マイグレート処理終了までのユーザー CPU 時間（単位：秒）。
SYSCPU	Explicit メモリーブロックの自動マイグレート予約の処理の開始から、マイグレート処理終了までのシステム CPU 時間（単位：秒）。
CAUSE	必ずMigrate と出力されます。Explicit メモリーブロックのマイグレート処理で出力したログであることを示します。

normal の場合（Explicit メモリーブロックのマイグレート処理時の Java ヒープあふれ）

出力形式

```
[ENS]ctime[EH: EH_USED_BF->EH_USED_AF(EH_TOTAL/EH_MAX),
ELAPSED secs][E/F/D: AC_NUM/FL_NUM/DA_NUM]
[DefNew::Eden: ED_USED_BF->ED_USED_AF(ED_TOTAL)]
[DefNew::Survivor: SV_USED_BF->SV_USED_AF(SV_TOTAL)]
[Tenured: TN_USED_BF->TN_USED_AF(TN_TOTAL)]
[target:EH_MIG_TRG/EH_MIG_DED/EH_MIG_LIV]
[User: USERCPU secs][Sys: SYSCPU secs][cause:CAUSE]
```

出力項目

出力項目	説明
ctime	Explicit メモリーブロックのマイグレート処理発生の日時。 拡張 verbosegc 機能で出力しているものと同一の時刻形式。 HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
EH_USED_BF	Explicit メモリーブロックのマイグレート処理前の、Explicit ヒープ利用済みサイズ（単位：キロバイト）。
EH_USED_AF	Java ヒープがあふれたあとの、Explicit ヒープ利用済みサイズ（単位：キロバイト）。
EH_TOTAL	Java ヒープがあふれたあとの、確保済み Explicit ヒープサイズ（単位：キロバイト）。
EH_MAX	Explicit ヒープ最大サイズ（単位：キロバイト）。
ELAPSED	Explicit メモリーブロックのマイグレート処理開始から、Java ヒープがあふれるまでの時間（単位：秒）。
AC_NUM	Explicit メモリーブロックのマイグレート処理後の、サブ状態がEnable である有効な Explicit メモリーブロック数。
FL_NUM	拡張用の出力項目であり、常に 0 を示します。
DA_NUM	Java ヒープがあふれたあとの、サブ状態がDisable である有効な Explicit メモリーブロック数。
ED_USED_BF	Explicit メモリーブロックのマイグレート処理前の、Eden 領域利用済みサイズ（単位：キロバイト）。
ED_USED_AF	Java ヒープがあふれたあとの、Eden 領域利用済みサイズ（単位：キロバイト）。
ED_TOTAL	Java ヒープがあふれたあとの、Eden 領域確保済みサイズ（単位：キロバイト）。
SV_USED_BF	Explicit メモリーブロックのマイグレート処理前の、Survivor 領域利用済みサイズ（単位：キロバイト）。
SV_USED_AF	Java ヒープがあふれたあとの、Survivor 領域利用済みサイズ（単位：キロバイト）。
SV_TOTAL	Java ヒープがあふれたあとの、Survivor 領域確保済みサイズ（単位：キロバイト）。
TN_USED_BF	Explicit メモリーブロックのマイグレート処理前の、Tenured 領域利用済みサイズ（単位：キロバイト）。
TN_USED_AF	Java ヒープがあふれたあとの、Tenured 領域利用済みサイズ（単位：キロバイト）。
TN_TOTAL	Java ヒープがあふれたあとの、Tenured 領域確保済みサイズ（単位：キロバイト）。
EH_MIG_TRG	Explicit メモリーブロックのマイグレート処理を行った Explicit ヒープの利用済みサイズ（単位：キロバイト）。
EH_MIG_DED	Java ヒープがあふれたあとの、それまでに Explicit メモリーブロックのマイグレート処理を行ったことによって減少した、Explicit ヒープの利用済みサイズ（単位：キロバイト）。常に 0K を出力します。
EH_MIG_LIV	Java ヒープがあふれたあとの、それまでに Explicit メモリーブロックのマイグレート処理を行ったことによって減少しなかった、Explicit ヒープの利用済みサイズ（単位：キロバイト）。Java ヒープあふれの発生原因となったオブジェクトのサイズは含まれません。

出力項目	説明
USERCPU	Explicit メモリーブロックのマイグレート処理開始から、Java ヒープがあふれるまでのユーザー CPU 時間（単位：秒）。
SYSCPU	Explicit メモリーブロックのマイグレート処理開始から、Java ヒープがあふれるまでのシステム CPU 時間（単位：秒）。
CAUSE	必ずMigratingと出力されます。Explicit メモリーブロックのマイグレート処理時の Java ヒープあふれで出力したログであることを示します。

normal の場合（明示管理ヒープ自動配置設定ファイルオープンエラー）

出力形式

```
[ENA]ctime failed to open file. [file=FILENAME]
```

出力項目

出力項目	説明
ctime	明示管理ヒープ自動配置設定ファイルのオープンに失敗した日時。拡張 verbosegc 機能で出力しているものと同一の時刻形式。HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
FILENAME	ファイルのオープンに失敗した自動配置設定ファイルの名前(ディレクトリー名を含まない)。

normal の場合（明示管理ヒープ自動配置設定ファイルパースエラー）

出力形式

```
[ENA]ctime parsed error line. [file=FILENAME line=LINENO]
```

出力項目

出力項目	説明
ctime	明示管理ヒープ機能自動配置設定ファイルのパースに失敗した日時。拡張 verbosegc 機能で出力しているものと同一の時刻形式。HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
FILENAME	ファイルのパースに失敗した自動配置設定ファイルの名前(ディレクトリー名を含まない)。
LINENO	パースに失敗した行数。

normal の場合（明示管理ヒープ自動配置エラー）

出力形式

```
[ENA]ctime creation
CLASS_LIST class's object in explicit memory is failed.
[target=CLASS_METHOD
detail=MESSAGE]
```

出力項目

出力項目	説明
ctime	明示管理ヒープ機能が明示管理ヒープ配置化に失敗した日時。拡張 verbosegc 機能で出力しているものと同一の時刻形式。HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
CLASS_LIST	明示管理ヒープ上へ配置しようとしたオブジェクトの完全限定クラス名のリスト。リストが空の場合があります。
CLASS_METHOD	明示管理ヒープ配置化に失敗したクラスの完全限定名。より詳細な失敗個所を示すメソッド名も出力することがあります。
MESSAGE	明示管理ヒープ配置化に失敗した原因を示す詳細メッセージ。

normal の場合（明示管理ヒープ自動配置のスキップ）

出力形式

```
[ENA]ctime creation class's object in explicit memory is skipped. [
detail = MESSAGE]
```

出力項目

出力項目	説明
ctime	明示管理ヒープ機能が明示管理ヒープ配置化をスキップした日時。拡張 verbosegc 機能で出力しているものと同一の時刻形式。HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
MESSAGE	明示管理ヒープ配置化をスキップした原因を示す詳細メッセージ。

normal の場合（明示管理ヒープ機能適用除外クラス指定機能の設定ファイルオープンエラー）

出力形式

```
[ENO]ctime failed to open file. [TYPE] [file=FILENAME]
```

出力項目

出力項目	説明
ctime	明示管理ヒープ機能適用除外クラス指定機能の設定ファイルのオープンに失敗した日時。拡張 verbosegc 機能で出力しているものと同一の時刻形式。 HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
TYPE	ファイルのオープンや読み込みに失敗した設定ファイルの種類を示します。 SYS:システムが設定している設定ファイル USR:ユーザーがオプションで指定したファイルパスの設定ファイル DEF:デフォルト配置場所にあるユーザー用の設定ファイル
FILENAME	ファイルのオープンに失敗した設定ファイルの名前(ディレクトリー名を含まない)。

normal の場合（明示管理ヒープ機能適用除外クラス指定機能の設定ファイルパースエラー ）

出力形式

```
[ENO]ctime parsed error line. [TYPE] [file=FILENAME line=LINENO]
```

出力項目

出力項目	説明
ctime	明示管理ヒープ機能適用除外クラス指定機能の設定ファイルのパースに失敗した日時。拡張 verbosegc 機能で出力しているものと同一の時刻形式。HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
TYPE	ファイルのパースに失敗した設定ファイルの種類を示します。 SYS:システムが設定している設定ファイル USR:ユーザーがオプションで指定したファイルパスの設定ファイル DEF:デフォルト配置場所にあるユーザー用の設定ファイル
FILENAME	ファイルのパースに失敗した設定ファイルの名前(ディレクトリー名を含まない)。
LINENO	パースに失敗した行数。

verbose の場合（Explicit メモリーブロックの初期化 ）

出力形式

```
[EVO]ctime>[Created]["EM_NAME" eid=EID(EM_PTR)/EM_TYPE]
```

出力項目

出力項目	説明
ctime	Explicit メモリーブロック初期化の日時。拡張 verbosegc 機能で出力しているものと同一の時刻形式。HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
EM_NAME	初期化した Explicit メモリーブロック名。 Explicit メモリーブロック名にマルチバイト文字を含む場合の出力内容は未定義(通常文字化け)とします。
EID	初期化した Explicit メモリーブロックの ID。
EM_PTR	Explicit メモリーブロック内部状態を示す値。
EM_TYPE	Explicit メモリーブロックの Java VM 内部での種別を示します。

verbose の場合（Explicit メモリーブロックの初期化失敗 ）

出力形式

```
[EVO]ctime[Creation failed][EH: EH_USED(EH_GARB)/EH_TOTAL/EH_MAX]
[E/F/D: AC_NUM/FL_NUM/DA_NUM][Thread: TH_PTR]
[EVO][Thread: TH_PTR] at FRAMESOURCE
...
```

出力項目

出力項目	説明
ctime	Explicit メモリーブロック初期化失敗の日時。拡張 verbosegc 機能で出力しているものと同一の時刻形式。HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。

出力項目	説明
EH_USED	Explicit メモリーブロック初期化失敗時の、Explicit ヒープ利用済みサイズ。
EH_GARB	拡張用の出力項目であり、Explicit ヒープの内部状態を示す（単位：キロバイト）。
EH_TOTAL	Explicit メモリーブロック初期化失敗時の Explicit ヒープ確保済みサイズ（単位：キロバイト）。
EH_MAX	Explicit ヒープ最大サイズ（単位：キロバイト）。
AC_NUM	Explicit メモリーブロック初期化失敗時の、サブ状態が`Enable` である有効な Explicit メモリーブロック数。
FL_NUM	拡張用の出力項目であり、常に 0 を示します。
DA_NUM	Explicit メモリーブロック初期化失敗時の、サブ状態が`Disable` である有効な Explicit メモリーブロック数。
TH_PTR	Explicit メモリーブロックの初期化に失敗したスレッドのスレッド ID。スレッドダンプに出力する tid と同一。
FRAME	Explicit メモリーブロック初期化失敗時のスタックトレース中の 1 フレーム。完全クラス名とメソッド名がピリオド（.）で区切って出力されます。
SOURCE	*FRAME* のメソッドが記述されているソースファイル名とスタックトレースと一致する行番号がコロン（:）で区切って出力されます。 ネイティブメソッドの場合は、`(Native Method)`、ソースファイル名が取得できない場合は、`(Unknown Source)`とします。

`verbose` の場合（Explicit メモリーブロックサブ状態の FreeList 化 ）

出力形式

```
[EVO]ctime[Alloc failed(FreeList)]
[EH: EH_USED(EH_GARB)/EH_TOTAL/EH_MAX]
[E/F/D: AC_NUM/FL_NUM/DA_NUM][cause:CAUSE]
["EM_NAME" eid=EID/EM_TYPE: EM_USED(EM_GARB)/EM_TOTAL]
[Thread: TH_PTR]
[EVO][Thread: TH_PTR] at FRAMESOURCE
...
```

注

「`[Thread: TH_PTR] [EVO][Thread: TH_PTR] at FRAMESOURCE`」は`New` の場合にだけ出力されます。

出力項目

出力項目	説明
ctime	*EID* が示す Explicit メモリーブロックのサブ状態が、FreeList 化した日時。拡張 verbosegc 機能で出力しているものと同一の時刻形式。`HitachiOutputMilliTime` オプションが有効な場合、ミリ秒単位まで出力されます。
EH_USED	*EID* が示す Explicit メモリーブロックのサブ状態が FreeList 化したときの Explicit ヒープ利用済みサイズ（単位：キロバイト）。
EH_GARB	拡張用の出力項目であり、Explicit ヒープの内部状態を示します。

出力項目	説明
EH_TOTAL	*EID* が示す Explicit メモリーブロックのサブ状態が FreeList 化したときの Explicit ヒープ確保済みサイズ（単位：キロバイト）。
EH_MAX	Explicit ヒープ最大サイズ（単位：キロバイト）。
AC_NUM	*EID* が示す Explicit メモリーブロックのサブ状態が FreeList 化した後の、サブ状態が`Enable`である有効な Explicit メモリーブロック数。
FL_NUM	拡張用の出力項目であり、常に 0 を示します。
DA_NUM	*EID* が示す Explicit メモリーブロックのサブ状態が FreeList 化した後の、サブ状態が`Disable`である有効な Explicit メモリーブロック数。
CAUSE	サブ状態の FreeList 化の原因となった処理。 `New` は、newInstance()などの Explicit ヒープへのオブジェクト直接生成。`GC` は Copy GC、`Full GC` は Full GC 時の Explicit ヒープへの移動処理が原因であったことを示します。
EM_NAME	サブ状態が FreeList になった Explicit メモリーブロック名。 Explicit メモリーブロック名に多バイト文字を含む場合の出力内容は未定義(通常文字化け)とします。また、Explicit メモリーブロックの初期化とほぼ同時に出力した場合や Java VM が内部で生成した Explicit メモリーブロックの場合は、`NULL` となることがあります。
EID	サブ状態が FreeList になった Explicit メモリーブロックの ID。
EM_TYPE	サブ状態が FreeList になった Explicit メモリーブロックの Explicit メモリーブロックの種別。 Explicit メモリーブロックの Java VM 内部での種別を示します。
EM_USED	サブ状態が FreeList になった Explicit メモリーブロックの利用済みサイズ（単位：キロバイト）。
EM_GARB	拡張用の出力項目であり、Explicit メモリーブロックの内部状態を示します。
EM_TOTAL	サブ状態が FreeList になった Explicit メモリーブロックの確保済みサイズ（単位：キロバイト）。
TH_PTR	サブ状態の FreeList 化の原因となった Explicit ヒープへの生成を行なったスレッドのスレッド ID。スレッドダンプに出力する tid と同一。 *CAUSE* が`New` の場合にだけ出力。
FRAME	サブ状態の FreeList 化の原因となった Explicit ヒープへの直接生成のスタックトレースの 1 フレーム。完全クラス名とメソッド名がピリオド（.）で区切って出力されます。 *CAUSE* が`New` の場合にだけ出力。
SOURCE	*FRAME* のメソッドが記述されているソースファイル名とスタックトレースと一致する行番号をがコロン（:）で区切って出力されます。 ネイティブメソッドの場合は、`(Native Method)`、ソースファイル名が取得できない場合は、`(Unknown Source)`とします。 *CAUSE* が`New` の場合にだけ出力されます。

`verbose` の場合（Explicit メモリーブロックサブ状態の Disable 化 ）

出力形式

```
[EVO]ctime[Alloc failed(Disable)]
[EH: EH_USED(EH_GARB)/EH_TOTAL/EH_MAX]
[E/F/D: AC_NUM/FL_NUM/DA_NUM][cause:CAUSE]
["EM_NAME" eid=EID/EM_TYPE: EM_USED(EM_GARB)/EM_TOTAL]
[Thread: TH_PTR]
[EVO][Thread: TH_PTR] at FRAMESOURCE
...
```

注

「*TH_PTR*] at *FRAMESOURCE*」はNew の場合にだけ出力されます。

出力項目

出力項目	説明
ctime	*EID* が示す Explicit メモリーブロックのサブ状態が Disable 化した日時。拡張 verbosegc 機能で出力しているものと同一の時刻形式。HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
EH_USED	*EID* が示す Explicit メモリーブロックのサブ状態が Disable 化したときの Explicit ヒープ利用済みサイズ（単位：キロバイト）。
EH_GARB	拡張用の出力項目であり、Explicit ヒープの内部状態を示します。
EH_TOTAL	*EID* が示す Explicit メモリーブロックのサブ状態が Disable 化したときの Explicit ヒープ確保済みサイズ（単位：キロバイト）。
EH_MAX	Explicit ヒープ最大サイズ（単位：キロバイト）。
AC_NUM	*EID* が示す Explicit メモリーブロックのサブ状態が Disable 化した後の、サブ状態がEnable である有効な Explicit メモリーブロック数。
FL_NUM	拡張用の出力項目であり、常に 0 を示します。
DA_NUM	*EID* が示す Explicit メモリーブロックのサブ状態が Disable 化した後の、サブ状態がDisable である有効な Explicit メモリーブロック数。
CAUSE	サブ状態の Disable 化の原因となった処理。 New は、newInstance()などの Explicit ヒープへのオブジェクト直接生成。GC は Copy GC、Full GC は Full GC 時の Explicit ヒープへの移動処理が原因であったことを示します。
EM_NAME	サブ状態がDisable になった Explicit メモリーブロック名。 Explicit メモリーブロック名に多バイト文字を含む場合の出力内容は未定義(通常文字化け)とします。また、Explicit メモリーブロックの初期化とほぼ同時に出力した場合や Java VM が内部で生成した Explicit メモリーブロックの場合は、NULL となることがあります。
EID	サブ状態がDisable になった Explicit メモリーブロックの ID。
EM_TYPE	サブ状態がDisable になった Explicit メモリーブロックの種別。 Explicit メモリーブロックの Java VM 内部での種別を示します。
EM_USED	サブ状態がDisable になった Explicit メモリーブロックの利用済みサイズ（単位：キロバイト）。
EM_GARB	拡張用の出力項目であり、Explicit メモリーブロックの内部状態を示します。

出力項目	説明
EM_TOTAL	サブ状態がDisable になった Explicit メモリーブロックの確保済みサイズ（単位：キロバイト）。
TH_PTR	サブ状態の Disable 化の原因となった Explicit ヒープへの生成を行なったスレッドのスレッド ID。スレッドダンプに出力する tid と同一。 *CAUSE* がNew の場合にだけ出力。
FRAME	サブ状態の FreeList 化の原因となった Explicit ヒープへの直接生成のスタックトレースの 1 フレーム。完全クラス名とメソッド名がピリオド（.）で区切って出力されます。 *CAUSE* がNew の場合にだけ出力。
SOURCE	*FRAME* のメソッドが記述されているソースファイル名とスタックトレースと一致する行番号がコロン（:）で区切って出力されます。 ネイティブメソッドの場合は、(Native Method)、ソースファイル名が取得できない場合は、(Unknown Source)とします。 *CAUSE* がNew の場合にだけ出力されます。

verbose の場合（Explicit メモリーブロックへのオブジェクト生成 ）

出力形式

```
[EVS]ctime[EH: EH_USED_BF->EH_USED_AF(EH_TOTAL/EH_MAX)]
[E/F/D: AC_NUM/FL_NUM/DA_NUM][cause:CAUSE]
["EM_NAME" eid=EID/EM_TYPE: EM_USED_BF->EM_USED_AF(EM_TOTAL)]
```

出力項目

出力項目	説明
ctime	オブジェクト生成の日時。拡張 verbosegc 機能で出力しているものと同一の時刻形式。HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
EH_USED_BF	オブジェクト生成前の、Explicit ヒープ利用済みサイズ（単位：キロバイト）。
EH_USED_AF	オブジェクト生成後の、Explicit ヒープ利用済みサイズ（単位：キロバイト）。
EH_TOTAL	オブジェクト生成後の、確保済み Explicit ヒープサイズを示す（単位：キロバイト）。
EH_MAX	Explicit ヒープ最大サイズ（単位：キロバイト）。
AC_NUM	オブジェクト生成後の、サブ状態がEnable である有効な Explicit メモリーブロック数。
FL_NUM	拡張用の出力項目であり、常に 0 を示します。
DA_NUM	オブジェクト生成後の、サブ状態がDisable である有効な Explicit メモリーブロック数。
CAUSE	必ずNew と出力されます。Explicit メモリーブロックへのオブジェクト生成が、Java プログラムから行われたことを示します。
EM_NAME	オブジェクトを生成した Explicit メモリーブロック名。 Explicit メモリーブロック名に多バイト文字を含む場合の出力内容は未定義(通常文字化け)とします。また、Explicit メモリーブロックの初期化とほぼ同時に出力した場合や Java VM が内部で生成した Explicit メモリーブロックの場合は、NULL となることがあります。
EID	オブジェクトを生成した Explicit メモリーブロックの ID。

出力項目	説明
EM_TYPE	オブジェクトを生成した Explicit メモリーブロックの Explicit メモリーブロックの種別。 Explicit メモリーブロックの Java VM 内部での種別を示します。
EM_USED_BF	オブジェクト生成前の、生成先 Explicit メモリーブロックの利用済みサイズ（単位：キロバイト）。
EM_USED_AF	オブジェクト生成後の、生成先 Explicit メモリーブロックの利用済みサイズ（単位：キロバイト）。
EM_TOTAL	オブジェクト生成後の、生成先 Explicit メモリーブロックの確保済みサイズ（単位：キロバイト）。

verbose の場合（Explicit メモリーブロックへの移動の詳細 ）

出力形式

```
normalの場合（GC時のExplicitヒープの利用状況）の出力情報
[EVS]{["EM_NAME" eid=EID/EM_TYPE: EM_USED_BF->EM_USED_AF(EM_TOTAL)]}{1,5}
...
```

注

Explicit メモリーブロックの情報が 5 個出力されるごとに改行されます。

出力項目

出力項目	説明
GC 時の Explicit ヒープの利用状況の出力情報	normal の場合（GC 時の Explicit ヒープの利用状況）の出力項目と同じです。
EM_NAME	GC でのオブジェクト移動先となった Explicit メモリーブロック名。 Explicit メモリーブロック名に多バイト文字を含む場合の出力内容は未定義(通常文字化け)とします。また、Explicit メモリーブロックの初期化とほぼ同時に出力した場合や Java VM が内部で生成した Explicit メモリーブロックの場合は、NULL となることがあります。
EID	GC でのオブジェクト移動先となった Explicit メモリーブロックの ID。
EM_TYPE	GC でのオブジェクト移動先となった Explicit メモリーブロックの種別。 Explicit メモリーブロックの Java VM 内部での種別を示し、R またはA が出力されます。
EM_USED_BF	GC でのオブジェクト移動先となった Explicit メモリーブロックの、GC 前の利用済みサイズ（単位：キロバイト）。
EM_USED_AF	GC でのオブジェクト移動先となった Explicit メモリーブロックの、GC 後の利用済みサイズ（単位：キロバイト）。
EM_TOTAL	GC でのオブジェクト移動先となった Explicit メモリーブロックの、GC 後の確保済みサイズ（単位：キロバイト）。

verbose の場合（Explicit メモリーブロック解放処理の詳細）

出力形式

```
normalの場合（Explicitメモリーブロック解放処理）の出力情報
[EVS]{["EM_NAME" eid=EID/EM_TYPE: EM_TOTAL]}{1,5}
...
```

注

Explicit メモリーブロックの情報が 5 個出力されるごとに改行されます。

出力項目

出力項目	説明
normal の場合（Explicit メモリーブロック解放処理）の出力情報	normal の場合（Explicit メモリーブロック解放処理）の出力項目と同じです。
EM_NAME	解放した Explicit メモリーブロック名。 Explicit メモリーブロック名に多バイト文字を含む場合の出力内容は未定義(通常文字化け)とします。
EID	解放した Explicit メモリーブロックの ID。
EM_TYPE	解放した Explicit メモリーブロックの種別。 Explicit メモリーブロックの Java VM 内部での種別を示します。
EM_TOTAL	解放した Explicit メモリーブロックの確保済みだったサイズ(解放サイズ)（単位：キロバイト)。

verbose の場合（ファイナライザーによる Explicit メモリーブロックの解放予約）

出力形式

```
[EVO]ctime[Finalized]["EM_NAME" eid=EID/EM_TYPE: EM_TOTAL]
```

出力項目

出力項目	説明
ctime	解放予約の日時。拡張 verbosegc 機能で出力しているものと同一の時刻形式。 HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
EM_NAME	解放予約した Explicit メモリーブロック名。 Explicit メモリーブロック名に多バイト文字を含む場合の出力内容は未定義(通常文字化け)とします。
EID	解放予約した Explicit メモリーブロックの ID。
EM_TYPE	解放予約した Explicit メモリーブロックの種別。 Explicit メモリーブロックの Java VM 内部での種別を示します。
EM_TOTAL	解放予約した Explicit メモリーブロックの確保済みだったサイズ(解放サイズ)（単位：キロバイト)。

verbose の場合（明示管理ヒープ自動配置化）

出力形式

```
[EVA]ctime creation in explicit memory is succeeded. [class=CLASSNAME]
```

出力項目

出力項目	説明
ctime	明示管理ヒープ機能が指定のクラスの明示管理ヒープ配置化に成功した日時。拡張 verbosegc 機能で出力しているものと同一の時刻形式。HitachiOutputMilliTime オプションが有効な場合、ミリ秒単位まで出力されます。
CLASSNAME	明示管理ヒープ配置化に成功したクラスの完全修飾クラス名。

debug の場合（Explicit メモリーブロック解放による Java ヒープへのオブジェクト移動）

出力形式

```
[EDO][eid=EID: Reference to REFED_NAME(REFED_PTR), total R_SIZE]
[EDO]   REF_NAME(REF_PTR)REF_GEN
```

出力項目

出力項目	説明
EID	Explicit メモリーブロック解放処理時に、解放対象 Explicit ヒープ以外から参照されているオブジェクトを持つ Explicit メモリーブロックの ID。
REFED_NAME	Explicit メモリーブロック解放処理時に、解放対象 Explicit ヒープ以外のオブジェクト(*REF_NAME*(*REF_PTR*))から参照されているオブジェクトの完全クラス名。
REFED_PTR	*REFED_NAME* が示すオブジェクトのメモリーアドレス(Java ヒープへの移動前)。
R_SIZE	*REF_NAME*(*REF_PTR*)からの参照によって、Java ヒープへ戻ることになるオブジェクトの総サイズ。 *REF_NAME*(*REF_PTR*)から間接参照している、解放対象 Explicit メモリーブロック内オブジェクトを含めた値（単位：キロバイト）。
REF_NAME	*REFED_NAME*(*REFED_PTR*)への参照を持つオブジェクトの完全クラス名。ただし、*REFED_NAME*(*REFED_PTR*)への参照が、スタックや Java VM 内部からの場合は、JVM とします。
REF_PTR	*REF_NAME* のメモリーアドレス。
REF_GEN	*REF_NAME*(*REF_PTR*)の属する領域名(または世代名)。 Explicit メモリーブロックの場合は、eid 出力されます。スタックや Java VM 内部から参照されている場合はJVM とします。 New 領域については、マイナー GC にシリアルガーベジコレクターを使用している場合は DefNew となります。

debug の場合（Explicit メモリーブロックの初期化の詳細）

出力形式

```
verboseの場合（Explicitメモリーブロックの初期化 ）の出力情報
[Thread: TH_PTR]
[EDO][Thread: TH_PTR] at FRAMESOURCE
...
```

出力項目

出力項目	説明
verbose の場合（Explicit メモリーブロックの初期化 ）の出力情報	verbose の場合（Explicit メモリーブロックの初期化 ）の出力項目と同じです。
TH_PTR	Explicit メモリーブロックを初期化したスレッドのスレッド ID。スレッドダンプに出力する tid と同じです。
FRAME	Explicit メモリーブロック初期化時のスタックトレース中の 1 フレーム。完全クラス名とメソッド名がピリオド（.）で区切って出力されます。
SOURCE	*FRAME* のメソッドが記述されているソースファイル名とスタックトレースと一致する行番号がコロン（:）で区切って出力されます。 ネイティブメソッドの場合は、(Native Method)、ソースファイル名が取得できない場合は、(Unknown Source)とします。

debug の場合（Explicit メモリーブロックのマイグレート処理の詳細）

出力形式

normalの場合（Explicitメモリーブロックのマイグレート処理）の出力情報
[EDO][migrate:(*EID_DEL*{,*EID_DEL*}*|)/(*EID_MBF*{,
EID_MBF}*->*EID_MAF*|)/(*EID_MIG*{,*EID_MIG*}*|)]

出力項目

出力項目	説明
normal の場合（Explicit メモリーブロックのマイグレート処理）の出力情報	normal の場合（Explicit メモリーブロックのマイグレート処理）の出力項目と同じです。
EID_DEL	Explicit メモリーブロックのマイグレート処理で解放される Explicit メモリーブロックのうち、オブジェクトの移動が発生しなかった Explicit メモリーブロックの ID。
EID_MBF	Explicit メモリーブロックのマイグレート処理で解放される Explicit メモリーブロックのうち、多対 1 のマイグレートが行われた Explicit メモリーブロックのマイグレート選択される前の ID。
EID_MAF	Explicit メモリーブロックのマイグレート処理で生成される Explicit メモリーブロックの ID。
EID_MIG	Explicit メモリーブロックのマイグレート処理で解放される Explicit メモリーブロックの ID。

スレッドダンプログ

Java VM のスレッドダンプ情報の構成を次の表に示します。

出力項目	説明
ヘッダー	日付および時刻、Java VM バージョン情報、起動コマンドラインを出力します。
システム設定	次の情報を出力します。 • JDK の実行環境のインストール場所を表す Java ホームパス • JDK を構成するライブラリーのインストールディレクトリーを表す Java DLL パス

出力項目	説明
	• システムクラスパス • java コマンドオプション
動作環境	次の情報を出力します。 • ホスト名 • OS バージョン • CPU 情報 • リソース情報
Java ヒープ情報	Java ヒープの各世代のメモリ使用状況を出力します。
C ヒープマップ情報	Java VM 自身の確保しているメモリの領域情報を出力します。
C ヒープサイズ情報	Java VM 自身の確保しているメモリのサイズ情報を出力します。
アプリケーション環境	次の情報を出力します。 • シグナルハンドラー • 環境変数
ライブラリー情報	ローディングされているライブラリーの情報を出力します。
スレッド情報 *スレッド 1* ： *スレッド n*	スレッドごとにスレッド情報を出力します。
Java モニターダンプ	Java モニターオブジェクトの一覧を表示します。
JNI グローバル参照情報	Java VM が保持している JNI のグローバル参照の数を出力します。
Explicit ヒープ情報(-XX:+HitachiUseExplicitMemory が有効な場合)	明示管理ヒープ機能使用時には、Java プロセスのクラスごとに次の情報を出力します。 • Explicit ヒープ全体の利用状況 • Explicit メモリーブロックごとの利用状況 また、明示管理ヒープ機能使用時に、`eheapprof` コマンドを実行すると、Explicit メモリーブロック内のオブジェクトの統計情報、および Explicit メモリーブロックの解放率情報を出力します。
クラス別統計情報	`jheapprof` コマンドで指定した Java プロセスのクラスごとに次の情報を出力します。 • インスタンスがメンバーとして持つインスタンスの合計サイズ、およびインスタンスの参照関係 • static メンバーが持つインスタンスの合計サイズ • Tenured 領域の増加原因となるオブジェクトのクラス、およびインスタンスの合計サイズ
フッター	スレッドダンプが終了した時刻を表示します。

各項目の出力形式を記載します。

ヘッダー

出力形式

```
EEE MMM dd hh:mm:ss yyyy

Full thread dump Java HotSpot(TM) VM type
(Oracleバージョン情報-Developer's Kit for Javaバージョン情報-ビルド年月日 mixed mode)
コマンドライン
```

注

EEE は曜日、*MMM* は月、*dd* は日を示します。

出力項目

出力項目	説明
VM type	Client VM、Server VM または 64-Bit Server VM。
Oracle バージョン情報	ベースとなる Oracle 製 JDK のバージョン。 25.20-b23。
Developer's Kit for Java バージョン情報	*Developer's Kit for Java バージョン情報*のバージョン。 HJDK1010ZZ。 (ZZ は修正番号。正規版はなし。)
ビルド年月日	製品をビルドした年月日。
コマンドライン	VM 起動時のコマンドライン。

システム設定

出力形式

```
System Properties
-----------------
Java Home Dir    : Developer's Kit for Java実行環境のインストールディレクトリー
Java DLL Dir     : Developer's Kit for Javaライブラリーのインストールディレクトリー
Sys Classpath    : システムクラスパス
User Args    :
コマンドオプション1
コマンドオプション2
…
```

出力項目

出力項目	説明
Developer's Kit for JavaJDK 実行環境のインストールディレクトリー	Developer's Kit for Java の実行環境のインストール場所を表す Java ホームパスが表示されます。
Developer's Kit for Java ライブラリーのインストールディレクトリー	Developer's Kit for Java を構成するライブラリーのインストールディレクトリーを表す Java DLL パスを表示されます。
システムクラスパス	システムクラスパスが出力されます。
コマンドオプション	Java コマンドオプションが出力されます。

動作環境

出力形式

Linux/EM64T 版では次のように出力されます。

```
Operating Environment
---------------------
Host    : ホスト名:IPアドレス
OS    : OSバージョン
CPU    : CPU種別, 利用可能CPU数/システム全体のCPU数 active

Resource Limits -
        RLIMIT_CPU              : プロセスで使用可能な秒数
        RLIMIT_FSIZE            : 最大ファイルサイズ(単位：バイト)
        RLIMIT_DATA             : malloc可能なサイズ(単位：バイト)
        RLIMIT_STACK            : スタックの最大サイズ(単位：バイト)
        RLIMIT_CORE             : coreの最大サイズ(単位：バイト)
        RLIMIT_RSS              : プロセスの常駐サイズ(単位：バイト)
        RLIMIT_NOFILE           : 最大のファイルディスクリプター値
        RLIMIT_AS               : プロセストータルの利用可能メモリー(単位：バイト)
        RLIMIT_NPROC            : プロセスの最大数
        RLIMIT_MEMLOCK          : ロック可能なメモリーサイズ(単位：バイト)
```

AIX 版では次のように出力されます。

```
Operating Environment
---------------------
Host    : ホスト名:IPアドレス
OS    : OSバージョン
CPU    : CPU種別※, 利用可能CPU数/システム全体のCPU数 active

Resource Limits -
        RLIMIT_CPU              : プロセスで使用可能な秒数
        RLIMIT_FSIZE            : 最大ファイルサイズ(単位：バイト)
        RLIMIT_DATA             : malloc可能なサイズ(単位：バイト)
        RLIMIT_STACK            : スタックの最大サイズ(単位：バイト)
        RLIMIT_CORE             : coreの最大サイズ(単位：バイト)
        RLIMIT_RSS              : プロセスの常駐サイズ(単位：バイト)
        RLIMIT_AS               : プロセストータルの利用可能メモリー(単位：バイト)
        RLIMIT_NOFILE           : 最大のファイルディスクリプター値
```

出力項目

Linux/EM64T 版の出力項目を次に示します。

出力項目	説明
ホスト名	マシン名。
IP アドレス	マシンの IP アドレス。複数ある場合は、コンマ（,）で区切って出力されます。
OS バージョン	次の情報を出力します。 *OS名 バージョン リリース* OS 名 Linux と出力されます。

出力項目	説明
	UNIX コマンドのuname -s-と同等の情報が出力されます。 バージョン バージョン情報。 UNIX コマンドのuname -v と同等の情報が出力されます。 リリース リリース情報。 UNIX コマンドのuname -r と同等の情報が出力されます。
CPU 種別	CPU 名が出力されます。 UNIX コマンドのuname -m と同等の情報が出力されます。
利用可能 CPU 数	利用可能な論理 CPU の数が出力されます。
システム全体の CPU 数	搭載している論理 CPU の数が出力されます。

AIX 版の出力項目を次に示します。

出力項目	説明
ホスト名	マシン名。
IP アドレス	マシンの IP アドレス。複数ある場合は、コンマ（,）で区切って出力されます。
OS バージョン	次の情報を出力します。 *OS名 バージョン リリース*
CPU 種別※	既知の CPU の場合：CPU 名が出力されます。 未知の CPU の場合：unknown (*ARCH* architecture:*VER*)と出力されます。 ARCH 次のどれかが出力されます。 ・Power Classic ・Power PC ・Intel IA64 ・unknown VER CPU のバージョン番号(_system_configuration.version)が 16 進数で出力されます。
利用可能 CPU 数	利用可能な論理 CPU の数が出力されます。
システム全体の CPU 数	搭載している論理 CPU の数が出力されます。

注※

AIX の CPU 種別について

AIX で出力する *CPU 種別* は、Power PC 4 やPower PC 5 のように CPU 名を出力します。しかし、今後開発される CPU については、現段階では CPU を識別するためのバージョン値が不明であるため、CPU 名を出すことができません。そこで CPU のバージョン値が不明の場合は、CPU 情報の出力を以下にすることで、CPU の種別を判断できる情報を出します。

```
CPU    : unknown (Power PC architecture:CPUバージョン値),
利用可能CPU数/システム全体のCPU数

CPUバージョン値     : _system_configuration.versionの値を16進数で表した値。
```

_system_configuration は AIX のシステム変数であり、_system_configuration.version はプロセッサーのバージョンを表します。_system_configuration は/usr/include/sys/systemcfg.h で宣言されています。

Java ヒープ情報

出力形式

```
Heap Status
-----------
def new generation   max max size, total capacity, used size
(max usage% used/max, total usage% used/total)
                     [bottom, commit addr, reserve addr)
  eden space capacity,  usage% used [bottom,
top, reserve addr)
  from space capacity,   usage% used [bottom,
top, reserve addr)
  to   space capacity,   usage% used [bottom,
top, reserve addr)
tenured generation   max max size, total capacity>, used size
(max usage% used/max, total usage% used/total)
                     [bottom, commit addr, reserve addr)
   the space capacity,  usage% used [bottom, top,
used block, reserve addr)
Metaspace       max max size, capacity capacity words,
committed committed size,
reserved reserve size, used size
(max usage% used/max, total usage% used/committed)
  class space    max max size, capacity capacity words,
committed committed size,reserved reserve size, used size
(max usage% used/max, total usage% used/committed)
                [bottom, top, commit addr,
reserve addr)
```

出力項目

出力項目	説明
max size	最大の容量（単位：キロバイト）。Metaspace については-XX:MaxMetaspaceSize が未指定の場合は Metaspace の最大値は無制限であるため、最大の容量は unlimited が出力されます。
capacity	現在の容量（単位：キロバイト）。
capacity words	Metaspace のコミット済みのメモリー領域からフリー領域を除いた合計サイズ（単位：キロバイト）。
committed size	Metaspace のコミット済みの合計メモリーサイズ（単位：キロバイト）。
reserve size	予約済みのメモリーサイズ（単位：キロバイト）。

出力項目	説明
size	使用中のメモリーサイズ（単位：キロバイト）。
max usage	最大の容量に対する使用率。Metaspace については-XX:MaxMetaspaceSize が未指定の場合は Metaspace の最大値は無制限であるため、最大の容量に対する使用率は-%が出力されます。
total usage	現在の容量に対する使用率。
bottom	領域先頭アドレス。
top	使用中領域の先頭アドレス。
commit addr	コミット済み領域の末尾アドレス。
reserve addr	予約済み領域の末尾アドレス。
usage	使用率。
used block	次の空きブロックの先頭アドレス。

Java VM 内部メモリーマップ情報

出力形式

```
JVM Internal Memory Map
-----------------------
メモリー確保関数      :address = 開始アドレス - 終了アドレス（ size: サイズ）
…
```

出力項目

出力項目	説明
メモリー確保関数	mmap()かmalloc()のどれかです。アドレスは 16 進数で表記されます。

Java VM 内部メモリーサイズ情報

出力形式

```
JVM Internal Memory Status
--------------------------
  Heap Size     :確保しているメモリーサイズ（単位：バイト）
  Alloc Size     :使用中のメモリーサイズ（単位：バイト）
  Free Size     :未使用のメモリーサイズ（単位：バイト）
```

アプリケーション環境

出力形式

```
Application Environment
-----------------------
Signal Handlers -
シグナル種別 : [シグナルハンドラーアドレス], sa_mask[0]=シグナルマスク, sa_flags=特殊フラグ
 :
```

```
Changed Signal Handlers -
シグナル種別 : [シグナルハンドラーアドレス], sa_mask[0]=シグナルマスク, sa_flags=特殊フラグ
 :

Environment Variables -
環境変数=値

Current Directory -
  /opt/Cosminexus/CC/server/… :カレントディレクトリー
```

出力項目

出力項目	説明
シグナル種別	/usr/include/sys/signal.h に定義されているシグナル名（例：SIGSEGV、SIGBUS など）。
シグナルハンドラーアドレス	シグナルハンドラーのアドレス。ライブラリー名＋オフセットという形式で表示されることもある。16 進数で出力されます。
シグナルマスク	sigaction()で取り出せる構造の sa_mask フィールド値。16 進数で出力されます。
特殊フラグ	sigaction()で取り出せる構造の sa_flags フィールド値。16 進数で出力されます。

ライブラリー情報

出力形式

AIX

```
Loaded Libraries
----------------
Dynamic libraries:
コマンド
    text    :開始アドレス-終了アドレス（サイズ）
    data    :開始アドレス-終了アドレス（サイズ）
ライブラリー
    text    :開始アドレス-終了アドレス（サイズ）
    data    :開始アドレス-終了アドレス（サイズ）
...
```

スレッド情報

出力形式

```
Stack Trace
-----------

"スレッド名" #スレッド識別子 daemon prio=優先度 os_prio=OS優先度 jid=ハッシュ値 tid=スレッドID
nid=native ID status [開始アドレス..終了アドレス][改行]
    java.lang.Thread.State: スレッドの現在のステータス

    stack=[スタック開始アドレス..YellowPageアドレス..
RedPageアドレス..スタック終了アドレス][改行]
    [user cpu time=ユーザー時間ms, kernel cpu time=カーネル時間ms]
[blocked count=ブロック回数, waited count=待機回数][改行]
```

```
        at クラス名.メソッド名(メソッド種別)
        ...
```

出力項目

出力項目	説明
スレッド名	Thread クラスのコンストラクターに指定されたスレッド名。
スレッド識別子	Java スレッドが作成されたときに生成される一意の数字。 java.lang.Thread.getId()で得られる値と同じです。
daemon	デーモンスレッドである場合にdaemon と出力されます。
優先度	Thread#setPriority で設定された優先度。
OS 優先度	OS レベルの優先度。優先順位が設定できない OS ではすべて0 が出力されます。
ハッシュ値	16 進 8 桁の数値。スレッドオブジェクトに対して、System.identityHashCode()を呼び出して得られる値と同一の値です。ただし、そのオブジェクトに対してハッシュ値が割りついていない場合には、<N/A>を出力します。非 Java スレッドに対しては出力しません。
スレッド ID	スレッドオブジェクトのメモリー上のアドレス。
native ID	OS レベルのスレッド ID。
status	スレッドのステータス。 次の値を出力します。 runnable 実行可能状態 waiting on condition notify 待ち waiting for monitor entry synchronized ロック取得待ち suspend またはsleeping 実行中断状態
スレッドの現在のステータス	スレッドの現在のステータスを表すメッセージが出力されます。メッセージの内容は java.lang.Thread.State 列挙型に対応します。非 Java スレッドに対しては出力しません。
スタック開始アドレス	スタック開始アドレス。16 進数で出力されます。非 Java スレッドに対しては出力しません。
YellowPage アドレス	スタック Yellow ガードページ先頭アドレス。16 進数で出力されます。非 Java スレッドに対しては出力しません。
RedPage アドレス	スタック Red ガードページ先頭アドレス。16 進数で出力されます。非 Java スレッドに対しては出力しません。
スタック終了アドレス	スタック終了アドレス。16 進数。非 Java スレッドに対しては出力しません。
開始アドレス	Java フレームの最高位スタックアドレス。16 進数で出力されます。
終了アドレス	JavaLock のある最高位スタックアドレス。16 進数で出力されます。

出力項目	説明
ユーザー時間	スレッド開始からのユーザー時間（単位：ミリ秒）。時間が取得できない環境の場合は出力しません。また、非 Java スレッドに対しても出力しません。 ユーザー時間の取得に失敗した場合は、次のようにunknown を表示します。 [user cpu time=unknown, kernel cpu time=*カーネル時間*ms]
カーネル時間	スレッド開始からのカーネル時間。単位はミリ秒。時間が取得できない環境の場合、出力しません。また、非 Java スレッドに対しても出力しません。
ブロック回数	スレッド開始からの、処理がブロックされた回数。非 Java スレッドに対しては出力しません。
待機回数	スレッド開始からの、処理が待ち状態にあった回数。非 Java スレッドに対しては出力しません。
クラス名	クラス名。
メソッド名	メソッド名。
メソッド情報	ネイティブメソッドの場合 Native Method Java メソッドで行番号付きでコンパイルされている場合 *ファイル名*：*行番号* Java メソッドで行番号無しでコンパイルされている場合 Unknown Source

Java モニターダンプ

出力形式

```
Java monitor
------------
ロックオブジェクト@ハッシュコード オーナー情報
    待機状態:待機スレッド数
        待機スレッド情報
```

出力項目

出力項目	説明
ロックオブジェクト	ロック対象オブジェクトのクラス名が出力されます。
ハッシュコード	Object.hashCode で得られるハッシュコード。
オーナー情報	オーナーがある場合 owner *スレッド名 スレッドオブジェクトアドレス* オーナーが無い場合 no owner
待機状態	synchronized ブロック/メソッド実行待ちの場合 waiting to enter

出力項目	説明
	notify 待ちの場合 `waiting to be notified`
待機スレッド情報	次のように出力されます。 *スレッド名 スレッドオブジェクトアドレス*

Raw モニターダンプ

出力形式

```
Raw monitor dump
----------------
モニター名 モニターアドレス owner "スレッド名" スレッドアドレス
    以下はJavaモニターダンプの出力項目と同じです。
```

出力項目

出力項目	説明
モニター名	ロック対象モニター名。
モニターアドレス	モニターのアドレス。
スレッド名	モニターのロック取得スレッド名。
スレッドアドレス	オーナースレッドのスレッドアドレス。

JNI グローバル参照情報

出力形式

```
JNI Information
---------------
JNI global references: JNIグローバル参照数
```

出力項目

出力項目	説明
JNI グローバル参照数	Java VM が保持しているグローバル参照の数が出力されます。

Explicit ヒープ情報(-XX:+HitachiUseExplicitMemory が有効な場合)

出力形式（Explicit メモリブロックのオブジェクト解放率情報が無効な場合）

```
Explicit Heap:
 max EH_MAX, total EH_TOTAL, used EH_USED, garbage EH_GARB
(EH_PER1 used/max, EH_PER2
used/total, EH_PER3 garbage/used), EM_NUMS spaces exist
 Explicit Memories(EM_MGR_PTR)・・・・・・・・・・・・・・・・・・・・・・・・・・・・※1
  "EM_NAME" eid=EID(EM_PTR)/EM_TYPE, total EM_TOTAL,
used EM_USED, garbage EM_GARB
(EM_PER1 used/total, EM_PER2 garbage/used) EM_STAT・・・・・・・・・・・・・※2
    deployed objects
    _________________Size__Instances__Class_________________
                ISIZE      INUM CNAME
```

```
                ...
               AISIZE    AINUM total・・・・・・・・・・・・※3
```

注※1

「Explicit Heap：」から「(*EM_MGR_PTR*)」までが Explicit ヒープ情報です。

注※2

「*EM_NAME*」から「*EM_STAT*」までが Explicit メモリーブロック情報です。

注※3

ISIZE、*INUM*、*CNAME*、*AISIZE*、*AINUM* は Explicit メモリブロック内のオブジェクト統計情報です。

出力形式の補足事項

- Explicit ヒープ情報と Explicit メモリブロック情報の間には、空行が 1 行あります。
- Explicit メモリブロック情報の出力順序(どの Explicit メモリブロックから出力するか)は未定義とします。
- *EM_NAME* の前には、半角 2 文字分の空白があります。
- deployed objects の前には、半角 4 文字分の空白があります。
- *ISIZE* は、_____Size_のe に行末をそろえます。
- *INUM* は、__Instances の最後のs に行末をそろえます。
- 最後の行には空行が 1 行あります。したがって、Explicit メモリブロックごとの出力の間には空行が 1 行あります。

出力形式（Explicit メモリブロックのオブジェクト解放率情報が有効な場合）

```
Explicit Heap Status
--------------------
 max EH_MAX, total EH_TOTAL, used EH_USED,
garbage EH_GARB (EH_PER1 used/max, EH_PER2
used/total, EH_PER3 garbage/used), EM_NUMS spaces exist

 Explicit Memories(EM_MGR_PTR)・・・・・・・・・・・・・・・・・・・・・・・・・・・※1

  "EM_NAME" eid=EID(EM_PTR)/EM_TYPE, total EM_TOTAL,
used EM_USED, garbage EM_GARB
(EM_PER1 used/total, EM_PER2 garbage/used, FL_BLOCKS blocks)
EM_STAT・・・・・・・・・・・・※2
    deployed objects
    _____________Size__Instances__FreeRatio__Class_____________
                 ISIZE      INUM    FRATIO※4 CNAME
                ...
               AISIZE    AINUM total・・・・・・・・・・・・※3
```

注※1

「Explicit Heap：」から「(*EM_MGR_PTR*)」までが Explicit ヒープ情報です。

注※2

「*EM_NAME*」から「*EM_STAT*」までが Explicit メモリーブロック情報です。

注※3

ISIZE、*INUM*、*CNAME*、*AISIZE*、*AINUM* は Explicit メモリブロック内のオブジェクト統計情報です。

注※4

FRATIO は Explicit メモリブロックのオブジェクト解放率情報です。

出力形式の補足事項

- *FRATIO* は、__FreeRatio__のo に行末をそろえます。

出力項目

Explicit ヒープ情報の出力項目を次に示します。

出力項目	説明
EH_MAX	Explicit ヒープの最大サイズ（単位：キロバイト）。
EH_TOTAL	確保済み Explicit ヒープサイズ（単位：キロバイト）。
EH_USED	利用済み Explicit ヒープサイズ（単位：キロバイト）。
EH_GARB	拡張用の出力項目であり、Explicit ヒープの内部状態を示します。
EH_PER1	Explicit ヒープ利用率(*EH_USED/EH_MAX*)。
EH_PER2	Explicit ヒープ利用率(*EH_USED/EH_TOTAL*)。
EH_PER3	拡張用の出力項目であり、Explicit ヒープの内部状態を示します。
EM_NUMS	有効な Explicit メモリーブロックの数。
EM_MGR_PTR	Explicit ヒープ制御のための内部情報があるメモリーアドレス。

Explicit メモリーブロック情報の出力形式を次に示します。

出力項目	説明
EM_NAME	Explicit メモリーブロック名。 Explicit メモリーブロック名に多バイト文字を含む場合の出力内容は未定義(通常文字化け)とします。 また、Explicit メモリーブロックの初期化とほぼ同時に出力した場合や Java VM が内部で生成した Explicit メモリーブロックの場合は、NULL となることがあります。
EID	Explicit メモリーブロックの ID。
EM_PTR	Explicit メモリーブロック内部構造があるメモリーアドレス。
EM_TYPE	Explicit メモリーブロックの種別。 出力される値を次に示します。 R | B | A
EM_TOTAL	Explicit メモリーブロックのメモリー確保済みサイズ（単位：キロバイト）。
EM_USED	Explicit メモリーブロックの利用済みサイズ（単位：キロバイト）。
EM_GARB	拡張用の出力項目で、Explicit メモリーブロックの内部状態を示します。

出力項目	説明
EM_PER1	Explicit メモリーブロック利用率(*EM_USED/EM_TOTAL*)がパーセント表記されます。
EM_PER2	拡張用の出力項目で、Explicit メモリーブロックの内部状態を示します。
FL_BLOCKS	拡張用の出力項目で、常に**0** を示します。
EM_STAT	Explicit メモリーブロックのサブ状態。

Explicit メモリブロック内のオブジェクト統計情報の出力形式を次に示します。

出力項目	説明
ISIZE	あるクラスをインスタンス化したオブジェクトの、Explicit メモリブロック内のサイズ。
INUM	あるクラスをインスタンス化したオブジェクトの、Explicit メモリブロック内の個数。
CNAME	*ISIZE* および *INUM* が示すクラスの完全クラス名。
AISIZE	Explicit メモリブロック内の全オブジェクトの合計サイズ。
AINUM	Explicit メモリブロック内の全オブジェクトの個数。

Explicit メモリブロックのオブジェクト解放率情報の出力形式を次に示します。

出力項目	説明
FRATIO	xplicit メモリブロックの自動解放処理で解放されたオブジェクトの割合（オブジェクト解放率）（単位：%)。 オブジェクト解放率=（自動解放処理前のクラスのオブジェクト数－自動解放処理後のクラスのオブジェクト数）／自動解放処理前のクラスのオブジェクト数×100 なお、オブジェクト解放率情報出力時に、自動解放処理の対象とならなかったExplicit メモリブロックには、「-」が出力されます。

クラス別統計情報

出力形式

```
Java Heap Profile
----------------------
_________________Size__Instances__Class_________________
   total_size       Instance_count   class_name
   total_size       Instance_count   class_name
...
```

出力項目

出力項目	説明
total_size	インスタンスサイズの合計(単位：バイト)。
Instance_count	インスタンス数。

出力項目	説明
class_name	クラス名。

フッター

出力形式

```
Full thread dump completed.  EEE MMM dd hh:mm:ss yyyy
```

注

EEE は曜日、*MMM* は月、*dd* は日を示します。

プロセス起動時ログ

出力形式

メッセージログと同じです。

出力項目

メッセージログと同じです。

WebSocket アクセスログ

WebSocket アクセスログは、asadmin ユーティリティーコマンドのset-log-attributes サブコマンドの ServerInstance.websocket_access_log.format パラメーターで出力フォーマットを指定できます。

出力形式

デフォルトの設定の場合、次のフォーマットで出力されます。

```
%TS% %IO% %OPCODE% %URI% %FIN% %CLOSEREASON% %PAYLOADDATALEN% %ROOTAP% %CLIENTAP%
```

出力項目

出力項目	説明	デフォルト設定時の出力
%TS%	WebSocket フレームの送受信時刻。*yyyy/mm/dd hh:mm:ss.sss GMT時差* の形式で出力されます。	○
%CLIENTADDR%	WebSocket 通信を実施しているサーバインスタンスに接続しているクライアントの IP アドレスおよびポート番号。	－
%SERVERADDR%	WebSocket 通信を実施しているサーバインスタンスの IP アドレスおよびポート番号。	－
%IO%	WebSocket フレームの送受信方向。 IN：サーバインスタンスが WebSocket フレームを受信したことを示します。 OUT：サーバインスタンスが WebSocket フレームを送信したことを示します。	○
%URI%	リクエスト URI。	○

出力項目	説明	デフォルト設定時の出力
%ROOTAP%	ルートアプリケーション情報。	○
%CLIENTAP%	クライアントアプリケーション情報。	○
%OPCODE%	WebSocket フレームの種別。 Text、Binary、Ping、Pong、またはClose が出力されます。	○
%FIN%	WebSocket フレームの終端を示す識別子。 分割されたメッセージを送受信する場合、最後の WebSocket フレームにFINAL（終端）が設定され、最後の WebSocket フレームを送受信するまではCONT（継続）が設定されます。 CONT：WebSocket フレームの継続 FINAL：WebSocket フレームの終端	○
%PAYLOADDATALEN%	ペイロードデータ長。	○
%MASK%	ペイロードデータのマスク有無。 MASK：マスクされる NOMASK：マスクされない	−
%SESSIONID%	WebSocket セッションの ID。	−
%MASKKEY%	ペイロードデータのマスクに使うキー情報。 ペイロードデータがマスクされない場合は-が出力されます。	−
%ISEXTENDED%	送受信中の WebSocket プロトコルの拡張有無。 BASE：基本プロトコル EXTENDED：拡張プロトコル	−
%RSV%	WebSocket プロトコル拡張時の識別子。 送受信中の WebSocket プロトコルが拡張される際に、拡張を解釈するための識別子がこの領域に設定されます。	−
%FRAMEMAINTYPE%	WebSocket フレームの主な分類。 Date：Data フレーム Control：制御フレーム	−
%CLOSEREASON%	WebSocket コネクションが切断された理由。	○
%PAYLOADDATA(*n*)%	ペイロードデータ。 ペイロードデータがテキスト形式の場合、最初の*n* 文字と最後の*n* 文字が出力されます。%PAYLOADDATA%と同時に指定することはできません。 *n*：1～32768	−
%PAYLOADDATA%	ペイロードデータ。 ペイロードデータがテキスト形式の場合、ペイロードデータの内容を出力します。ペイロードデータの文字数が 65536 を超えるときは、%PAYLOADDATA(*n*)%でn=32768 を指定した場合と同様に出力されます。 %PAYLOADDATA(*n*)% と同時に指定することはできません。	−

（凡例）

○：出力されます。

－：出力されません。

保守ログ

ログレベル

ログレベル	概要
1	製品で異常処理が行われた場合にメッセージを出力します。
2	レベル 1 のメッセージに加え、正常処理中のメッセージも出力します。

9.2.3 Web サーバのログの出力形式

Web サーバのリクエストログおよびアクセスログの出力形式を次に示します。

リクエストログ

出力形式

[*時刻*]（*サーバプロセスID*）client : hws <--（*クライアントIPアドレス*:*ポート番号*,*サーバIPアドレス*:*ポート番号*[R]）(*ルートアプリケーション情報*)

出力例

```
[Tue May 27 10:10:08.045 2014] (564) client : hws -->
(10.210.185.27:49222,10.210.185.27:80[A])
[Tue May 27 10:10:08.046 2014] (564) client : hws <--
(10.210.185.27:49222,10.210.185.27:80[R])(10.209.15.47/250/0x0000000000000001)
```

アクセスログ

アクセスログへのルートアプリケーション情報出力は、ログフォーマットに%{hws_ap_root}n を指定することで出力されます。

ログフォーマットを次のように指定した場合のアクセスログの出力例を示します。

```
LogFormat "%h %l %u %t ¥"%r¥" %>s %b ¥"%{Referer}i¥" ¥"%{User-Agent}i¥" %I %O" combinedio
LogFormat "%h %l %u %t ¥"%r¥" %>s %b ¥"%{Referer}i¥" ¥"%{User-Agent}i¥"" combined
LogFormat "%h %l %u %t ¥"%r¥" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
LogFormat "%h %l %u %t ¥"%r¥" %>s %b %P %{hws_ap_root}n %I %O %X %D ¥"%{Referer}i¥" ¥"%
{User-Agent}i¥"" hws_trace
LogFormat "%h %l %u %t ¥"%r¥" %>s %b %T %P %{hws_ap_root}n" hws_std
```

出力例

```
10.210.185.27 - - [27/May/2014:10:10:08.045 +0900] "GET / HTTP/1.0" 200 52 0.001 564
10.209.15.47/250/0x0000000000000001
```

WebSocket リクエストログ

WebSocket リクエストログの出力項目と、出力パターンごとの出力形式、および出力例について次に示します。

出力項目

出力項目	説明
ID	prefork MPM の場合：*プロセス ID* worker MPM の場合：*プロセス ID*:*スレッド ID*
クライアント IP:ポート番号	クライアントの IP アドレスおよびポート番号。
IP アドレス 1:ポート番号 1	クライアントとの接続で使用している Web サーバの IP アドレスおよびポート番号。
IP アドレス 2:ポート番号 2	バックエンドサーバとの接続で使用している Web サーバの IP アドレスおよびポート番号。
IP アドレス 3:ポート番号 3	バックエンドサーバの IP アドレスおよびポート番号。
ルート AP 情報	ルートアプリケーション情報。 ルートアプリケーション情報が取得できない場合、-が出力されます。
ステータスコード	ステータスコード。 ステータスコードを取得できない場合、-1 が出力されます。
-->	クライアントからバックエンドサーバ方向へのデータの流れ。
<--	バックエンドサーバからクライアント方向へのデータの流れ。
-X-	データの送受信でエラー、またはコネクション切断。
---	事象なし。

出力形式

クライアントからバックエンドへのデータ送信に成功した場合

[時刻]（ID）クライアントIP:ポート番号 --> IPアドレス1:ポート番号1 --> IPアドレス2:ポート番号2 --> IPアドレス3:ポート番号3（ ルートAP情報 ）

バックエンドからクライアントへのデータ送信に成功した場合（初回）

[時刻]（ID）クライアントIP:ポート番号 <-- IPアドレス1:ポート番号1 <-- IPアドレス2:ポート番号2 <-- IPアドレス3:ポート番号3（ ルートAP情報 ）(ステータスコード ）

バックエンドからクライアントへのデータ送信に成功した場合（2 回目以降）

[時刻]（ID）クライアントIP:ポート番号 <-- IPアドレス1:ポート番号1 <-- IPアドレス2:ポート番号2 <-- IPアドレス3:ポート番号3（ ルートAP情報 ）

クライアントからのデータ受信でエラーが起きた場合

```
[時刻](ID)クライアントIP:ポート番号 -X- IPアドレス1:ポート番号1 --> IPアドレス2:ポート番号2 --- IPアドレス3:ポート番号3( ルートAP情報 )
```

バックエンドへのデータ送信でエラーが起きた場合

```
[時刻](ID)クライアントIP:ポート番号 --> IPアドレス1:ポート番号1 --> IPアドレス2:ポート番号2 -X- IPアドレス3:ポート番号3( ルートAP情報 )
```

バックエンドからのデータ受信でエラーが起きた場合

```
[時刻](ID)クライアントIP:ポート番号 --- IPアドレス1:ポート番号1 <-- IPアドレス2:ポート番号2 -X- IPアドレス3:ポート番号3( ルートAP情報 )
```

クライアントへのデータ送信でエラーが起きた場合

```
[時刻](ID)クライアントIP:ポート番号 -X- IPアドレス1:ポート番号1 <-- IPアドレス2:ポート番号2 <-- IPアドレス3:ポート番号3( ルートAP情報 )
```

バックエンドとのコネクション切断を検知した場合

```
[時刻](ID)クライアントIP:ポート番号 --- IPアドレス1:ポート番号1 --- IPアドレス2:ポート番号2 -X- IPアドレス3:ポート番号3( ルートAP情報 )
```

クライアントとのコネクション切断を検知した場合

```
[時刻](ID)クライアントIP:ポート番号 --- IPアドレス1:ポート番号1 -X- IPアドレス2:ポート番号2 --- IPアドレス3:ポート番号3( ルートAP情報 )
```

出力例

時刻、および*ID*（プロセス ID またはスレッド ID）は省略しています。

```
10.210.185.27:49504 --> 10.196.211.205:443 --> 10.196.211.205:35307 --> 10.210.185.50:80
(10.196.211.205/24805/0x00000000000193b1)
10.210.185.27:49504 <-- 10.196.211.205:443 <-- 10.196.211.205:35307 <-- 10.210.185.50:80
(10.196.211.205/24805/0x00000000000193b1)(101)
10.210.185.27:49504 <-- 10.196.211.205:443 <-- 10.196.211.205:35307 <-- 10.210.185.50:80
(10.196.211.205/24805/0x00000000000193b1)
10.210.185.27:49504 --> 10.196.211.205:443 --> 10.196.211.205:35307 --> 10.210.185.50:80
(10.196.211.205/24805/0x00000000000193b1)
10.210.185.27:49504 <-- 10.196.211.205:443 <-- 10.196.211.205:35307 <-- 10.210.185.50:80
(10.196.211.205/24805/0x00000000000193b1)
10.210.185.27:49504 --> 10.196.211.205:443 --> 10.196.211.205:35307 --> 10.210.185.50:80
(10.196.211.205/24805/0x00000000000193b1)
10.210.185.27:49504 --- 10.196.211.205:443 --- 10.196.211.205:35307 -X- 10.210.185.50:80
(10.196.211.205/24805/0x00000000000193b1)
```

9.2.4 パフォーマンストレーサーのログの出力形式

パフォーマンストレーサーのメッセージは、%PRFSPOOL%¥log ディレクトリーに出力されます。

出力ディレクトリーを次に示します。ファイル名はprf_message+番号（01～32）です。

出力種別	出力ディレクトリー
PRF デーモン/PRF コマンド PRF 出力 API（CPRF.jar,CPRF.DLL）	$PRFSPOOL/log/*PRF 識別子*

ログファイルの切り替えはシフト方式であり、常にprf_message01 ファイルにログが出力されます。prf_message02～prf_message32 ファイルは、シフト先の古いログファイルです。

ログのシフト先は、シフト先のファイルが存在しない場合は 02,03…32 とprf_message02 から順番にログファイル名が選択され、prf_message01 ファイルがリネームされます。シフト先のファイルがすべて存在する場合は、ログファイルの更新時刻が古い順に削除またはリネームされます。

出力形式

日付・時刻:*メッセージID pid tid*:*メッセージテキスト*

出力項目

項目	説明
日付・時刻	トレースの取得日付・時刻が「aaa bbb dd hh:mm:ss.sss YYYY」の形式で出力されます。
メッセージ ID	メッセージ ID が出力されます。
pid	プロセス ID が出力されます。
tid	スレッド ID が出力されます。
メッセージテキスト	メッセージテキストが出力されます。メッセージテキスト中に改行が含まれることもあります。

9.2.5　アプリケーション開発環境のログの出力形式

アプリケーション開発環境が提供するサーバ構築エラーファイル(BuildEnvironment.err)の出力形式と出力項目を次に示します。

新規インストール時にデバッグ環境の構築に失敗した場合に出力されます。

出力形式

エラーメッセージ
エラー終了メッセージ
日付・時刻

出力項目

項目	説明
エラーメッセージ	エラーが発生したコマンドのエラーメッセージが出力されます。

項目	説明
エラー終了メッセージ	セットアップがエラー終了した旨のメッセージと、エラー発生コマンドの詳細情報が出力されます。
日付・時刻	コマンドの終了日時が *yyyy/mm/dd hh:mm:ss.ff* 形式で出力されます。

9.2.6 性能解析トレースファイルの出力形式

性能解析トレースでは、各機能レイヤーのトレース情報を収集します。

出力形式

ダンプ形式

```
PRF: レコード状態 Process: プロセスID Thread: スレッドID(ハッシュ値)
Trace: トレース通番
ProcessName: プロセス名
Event: イベントID Time: 年:月:日 時:分:秒 ミリ秒/マイクロ秒/ナノ秒
Rc: リターンコード
ClientAP:クライアントアプリケーションのIPアドレス
クライアントアプリケーションのプロセスID -
クライアントアプリケーションの通信番号
RootAP: ルートアプリケーションのIPアドレス
ルートアプリケーションのプロセスID -
ルートアプリケーションの通信番号
INT: インターフェース名 OPR: オペレーション名
Offset +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +a +b +c +d +e +f 0123456789abcdef
  ダンプ情報
```

CSV 形式

```
PRF,Process,Thread(hashcode),Trace,ProcessName,Event,Date,Time,
Time(msec/usec/nsec),Rc,ClientAP IP,ClientAP PID, ClientAP CommNo.,
RootAP IP,RootAP PID,RootAP CommNo.,INT,OPR,OPT,ASCII
レコード状態,プロセスID,スレッドID(ハッシュ値),トレース通番,プロセス名,
イベントID,年/月/日,時:分:秒,ミリ秒/マイクロ秒/ナノ秒,リターンコード,
クライアントアプリケーションのIPアドレス,
クライアントアプリケーションのプロセスID,
クライアントアプリケーションの通信番号,ルートアプリケーションのIPアドレス,
ルートアプリケーションのプロセスID,
ルートアプリケーションの通信番号,インターフェース名,オペレーション名,
ダンプ情報,ASCII文字情報
```

出力項目

項目	説明
PRF	そのプロセスレコードの状態。次のどちらかが出力されます。 Rec：レコードの状態は正常です。 ErrRec：レコードの状態は異常です。
Process	トレース情報を取得したプロセスのプロセス ID。10 桁以内の 10 進数が出力されます。

項目	説明
Thread	トレース情報を取得したプロセスのスレッド ID、およびスレッドのハッシュ値。スレッド ID、ハッシュ値は 18 桁以内の 16 進数が出力されます。
Trace	該当スレッドでのトレース通番。10 桁以内の 10 進数が出力されます。
ProcessName	プロセス名。32 文字以内のプロセスを表わす文字列が出力されます。
Event	イベント ID。6 桁の 16 進数が出力されます。
Time	トレース情報を取得した年月日、時間を出力します。次の形式で出力されます。 年/月/日 時:分:秒 ミリ秒/マイクロ秒/ナノ秒 出力例 `2000/02/12 13:43:44 363/200/000`
Rc	リターンコード。16 桁の 10 進数が出力されます。
ClientAP	クライアントアプリケーションの次の情報が出力されます。 IP アドレス プロセス ID 通信番号
RootAP	ルートアプリケーションの次の情報が出力されます。 IP アドレス プロセス ID 通信番号
INT	インターフェース名。33 文字以内の文字列※が出力されます。
OPR	オペレーション名。33 文字以内の文字列※が出力されます。
OPT	各イベントで取得した情報。514 文字以内のダンプ形式の取得情報が出力されます。
ASCII	各イベントで取得した情報。514 文字以内の ASCII 文字の取得情報が出力されます。

注※

インターフェース名、オペレーション名が 33 文字を超える場合は、次のどれかの方法で、33 文字で出力します。

先頭 32 文字＋*

先頭 16 文字＋*＋末尾 16 文字

*＋末尾 32 文字

9.3　トレース取得ポイントについて

各レイヤーのトレース取得ポイント と PRF トレース取得レベルについて説明します。

9.3.1　パフォーマンストレーサーのトレース取得ポイントについて

パフォーマンストレーサーのトレース取得ポイントや取得できる情報について説明します。

PRF トレースとは、Java EE サーバの決まった処理のポイントで出力されるトレース情報のことで、PRF トレース取得レベルを設定することで出力できるバイナリー形式のファイルです。

なお、PRF トレースでは、一連の処理をトレースできるようにするため、イベント単位の一連の処理に一貫したキーを設定して管理しています。イベント内のトレース情報が出力されるポイント（トレース取得ポイント）で出力されるトレースには、キー情報を付加します。キー情報を次の表に示します。

表 9-1　PRF トレースのキー情報

項番	PRF トレースのキー情報	説明
1	ルートアプリケーション情報	複数のプロセス間にわたる処理シーケンスで、一意の値として保持しなければならない情報です。 Application Server がシーケンスの起点であると識別できる個所で、ルートアプリケーション情報を付与します。例えば、アプリケーションクライアントから接続する場合、アプリケーションクライアント開始時に付与し、Web ブラウザーなど他のクライアントから接続する場合、Web サーバでのリクエスト受付時に付与します。
2	クライアントアプリケーション情報	クライアント・サーバ間通信の処理のシーケンスで、一意の値として保持しなければならない情報です。 Application Server がクライアント・サーバ間通信の起点であると識別できる個所で、クライアントアプリケーション情報を付与します。 例えば、Web サービス呼び出し時の HTTP 通信、EJB 呼び出し時の RMI/IIOP 通信などの通信電文の送信処理直前で付与します。

Web Server のログファイル（リクエストログ、アクセスログ）に、ルートアプリケーション情報が出力されるため、Web Server のログファイルと性能解析トレースファイル※を突き合わせて調査できます。

注※

PRF トレースをテキスト（CSV）形式に編集したファイルです。

機能レイヤーごとのトレース取得ポイントを次の表に示します。

表 9-2　トレース取得ポイント

項番	機能レイヤー	イベント ID※
1	Web コンテナー	0xB100～0xB101

項番	機能レイヤー	イベント ID※
		0xB200～0xB203
2	EJB コンテナー	0xB300～0xB307 0xBB00～0xBB03
3	JNDI	0xB400～0xB401
4	JTA	0xB500～0xB507
5	JDBC	0xB600～0xB665
6	JSF	0xB700～0xB70D
7	JMS	0xB800～0xB81F
8	JAX-RS	0xB900～0xB905
9	JAX-WS	0xBA00～0xBA0B
10	Concurrency Utilities	0xBC00～0xBC03
11	WebSocket	0xBD00～0xBD47

注※

各処理での PRF トレースの出力ポイントと位置情報です。

メモ

WebSocket のクライアントエンドポイントを利用した Java EE サーバのプロセス間通信

- クライアントエンドポイントに対する WebSocket 通信の送受信や、クライアントエンドポイントからの WebSocket 標準 API 呼び出しに対しては、PRF トレースが出力されません。
- クライアントエンドポイントからサーバエンドポイントに対してアプリケーション情報は引き継がれません。そのため、クライアントエンドポイントが Application Server 内のプロセス（Java EE サーバやアプリケーションクライアント）であっても、プロセス間でアプリケーション情報は紐づけられません。

関連項目

- 9.3.2　Web コンテナーのトレース取得ポイント
- 9.3.3　EJB コンテナーのトレース取得ポイント
- 9.3.4　JNDI のトレース取得ポイント
- 9.3.5　JTA のトレース取得ポイント
- 9.3.6　JDBC のトレース取得ポイント
- 9.3.7　JSF のトレース取得ポイント
- 9.3.8　JMS のトレース取得ポイント
- 9.3.9　JAX-RS のトレース取得ポイント

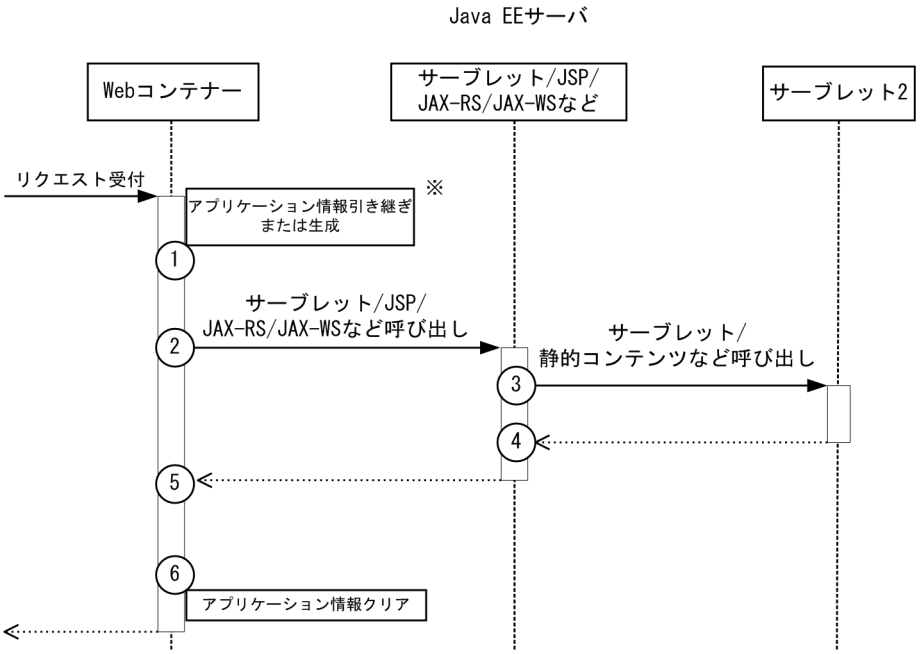
9.3.2　Web コンテナーのトレース取得ポイント

Web コンテナーでのトレース取得ポイントの詳細について説明します。

同期処理の場合

Web コンテナーでのトレースの取得ポイントを次に示します。

図 9-1　Web コンテナーのトレース取得ポイント（同期処理の場合）



（凡例）

(n)：トレース取得ポイントを示します。

> **メモ**
>
> この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

注※

リクエストヘッダーにアプリケーション情報が含まれている場合は引き継ぎます。

含まれていない場合はアプリケーション情報を生成します。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

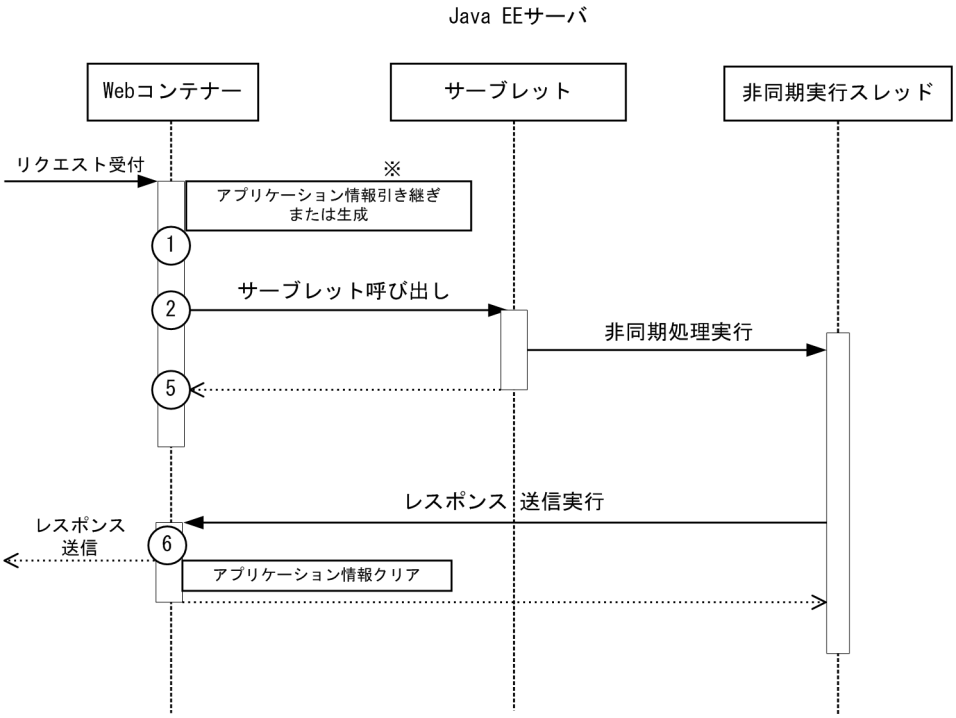
表 9-3　Web コンテナーのトレース取得ポイントの詳細（同期処理の場合）

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB100	1	標準レベル	リクエスト取得・リクエストヘッダー解析完了直後	HTTP メソッド名	URI	-
0xB101	6	標準レベル	リクエスト処理完了直後	HTTP メソッド名	URI	正常時 ステータスコード 異常時 ステータスコード：例外名
0xB200	2、3	標準レベル	サーブレット/JSP/JAX-RS/JAX-WS などの呼び出し直前	JSP 以外の場合 クラス名 JSP の場合 コンテキストルートを起点とする JSP ファイルパス	-	セッション ID
0xB201	4、5	標準レベル	サーブレット/JSP/JAX-RS/JAX-WS などの処理完了直後	JSP 以外の場合 クラス名 JSP の場合 コンテキストルートを起点とする JSP ファイルパス	-	正常時 セッション ID 異常時 セッション ID：例外名
0xB202	3	標準レベル	静的コンテンツ呼び出し直前（DefaultServlet）	-	コンテキストルート名	セッション ID
0xB203	4	標準レベル	静的コンテンツ処理完了直後（DefaultServlet）	-	コンテキストルート名	正常時 セッション ID 異常時 セッション ID：例外名

非同期処理の場合

サーブレットの非同期処理でのレスポンス送信時の、Web コンテナーでのトレースの取得ポイントを次に示します。

図 9-2　Web コンテナーの取得ポイント（非同期処理の場合）



（凡例）

(n)：トレース取得ポイントを示します。

> **メモ**
>
> この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

注※

リクエストヘッダーにアプリケーション情報が含まれている場合は引き継ぎます。

含まれていない場合はアプリケーション情報を生成します。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-4　Web コンテナーのトレース取得ポイントの詳細（非同期処理の場合）

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB100	1	標準レベル	リクエスト取得・リクエストヘッダー解析完了直後	HTTP メソッド名	URI	-
0xB101	6	標準レベル	リクエスト処理完了直後	HTTP メソッド名	URI	正常時 ステータスコード

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
						異常時 ステータスコード：例外名
0xB200	2	標準レベル	サーブレット/JSP/JAX-RS/JAX-WS などの呼び出し直前	JSP 以外の場合 クラス名 JSP の場合 コンテキストルートを起点とする JSP ファイルパス	-	セッション ID
0xB201	5	標準レベル	サーブレット/JSP/JAX-RS/JAX-WS などの処理完了直後	JSP 以外の場合 クラス名 JSP の場合 コンテキストルートを起点とする JSP ファイルパス	-	正常時 セッション ID 異常時 セッション ID：例外名

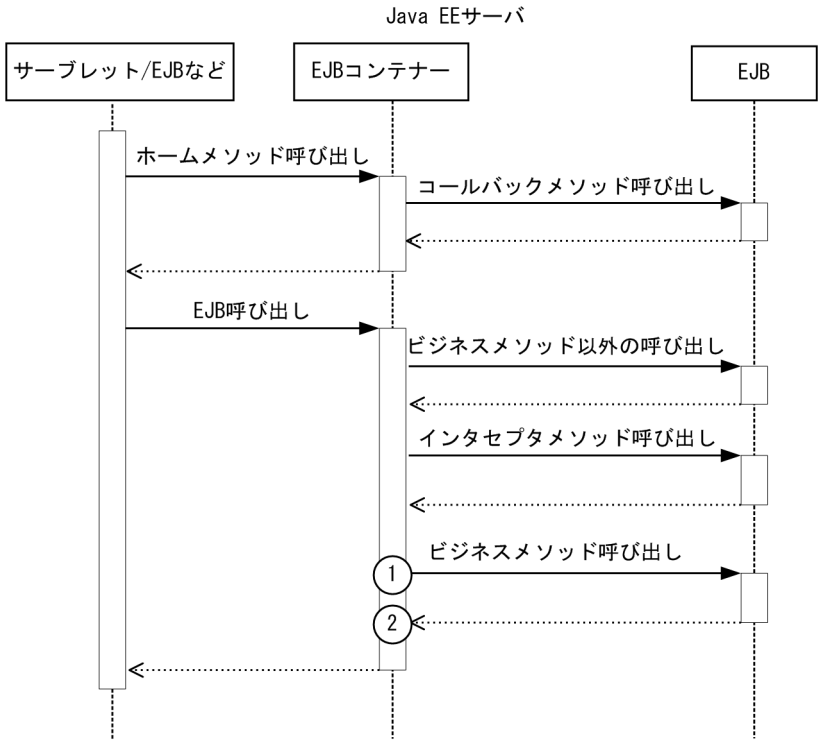
9.3.3 EJB コンテナーのトレース取得ポイント

EJB コンテナーでのトレース取得ポイントの詳細について説明します。

Session Bean および Entity Bean（ローカル呼び出し）の場合

Session Bean および Entity Bean（ローカル呼び出し）でのトレースの取得ポイントを次に示します。

図 9-3　Session Bean および Entity Bean（ローカル呼び出し）の取得ポイント



（凡例）

(n)：トレース取得ポイントを示します。

メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-5　トレース取得ポイントの詳細

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB300	1	標準レベル	EJB コンテナーが EJB のビジネスメソッドを呼び出す直前	EJB の実装クラス名	メソッド名(引数の数)	-
0xB301	2	標準レベル	EJB のビジネスメソッドの呼び出し直後	EJB の実装クラス名	メソッド名(引数の数)	正常時 - 異常時 例外名※

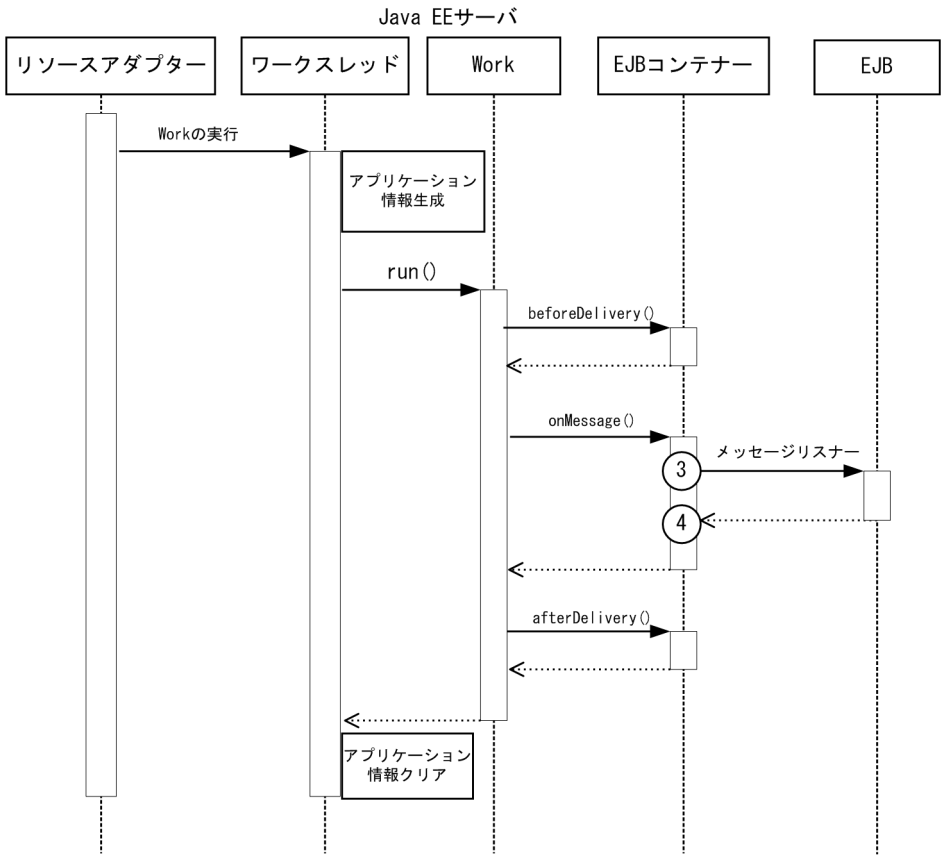
注※

トレース取得ポイントで取得した例外が java.lang.reflect.InvocationTargetException だった場合、java.lang.reflect.InvocationTargetException が保持している例外の例外名が出力されます。

Message Driven Bean の場合

Message Driven Bean での、トレースの取得ポイントを次に示します。

図 9-4　Message Driven Bean のトレース取得ポイント



（凡例）

(n)：トレース取得ポイントを示します。

> **メモ**
>
> この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-6　Message Driven Bean のトレース取得ポイントの詳細

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB302	3	標準レベル	Message Driven Bean のメッセージリスナー呼び出し直前	EJB の実装クラス名	メソッド名(引数の数)	-
0xB303	4	標準レベル	Message Driven Bean のメッセージリスナー呼び出し直後	EJB の実装クラス名	メソッド名(引数の数)	正常時 - 異常時 例外名※

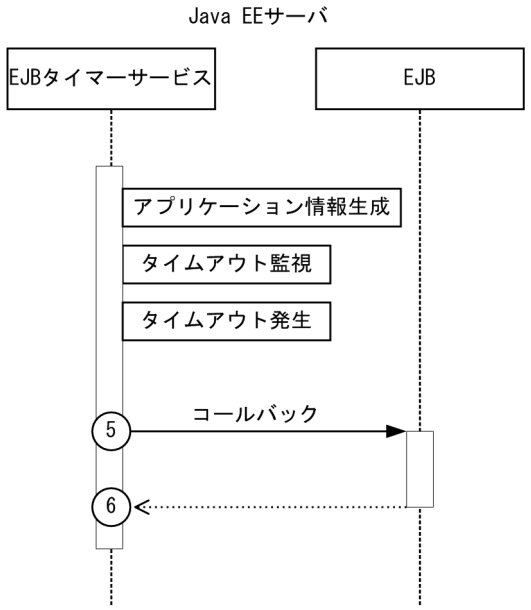
注※

トレース取得ポイントで取得した例外が java.lang.reflect.InvocationTargetException だった場合、java.lang.reflect.InvocationTargetException が保持している例外の例外名が出力されます。

Timer Service の場合

Timer Service でのトレースの取得ポイントを次に示します。

図 9-5　Timer Service のトレース取得ポイント



（凡例）

(n)：トレース取得ポイントを示します。

> **メモ**
>
> この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-7　Timer Service のトレース取得ポイントの詳細

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB304	5	標準レベル	EJB タイマーサービスのコールバックメソッド呼び出し直前	EJB の実装クラス名	メソッド名(引数の数)	-
0xB305	6	標準レベル	EJB タイマーサービスのコールバックメソッド呼び出し直後	EJB の実装クラス名	メソッド名(引数の数)	正常時 - 異常時 例外名※

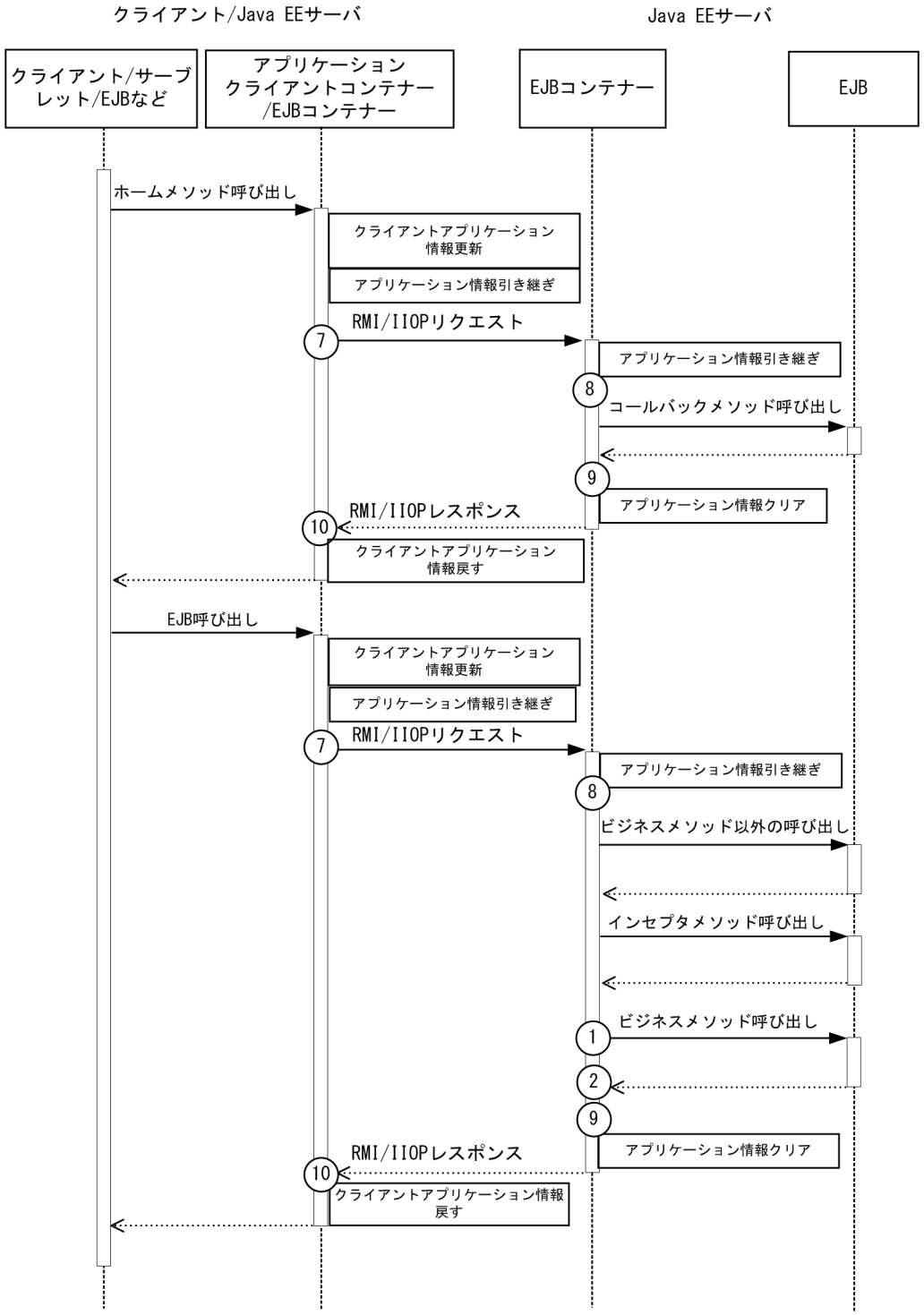
注※

トレース取得ポイントで取得した例外が java.lang.reflect.InvocationTargetException だった場合、java.lang.reflect.InvocationTargetException が保持している例外の例外名が出力されます。

Session Bean および Entity Bean（リモート呼び出し）の場合

Session Bean および Entity Bean（リモート呼び出し）でのトレースの取得ポイントを次に示します。

図 9-6　Session Bean および Entity Bean（リモート呼び出し）のトレース取得ポイント



メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-8　Session Bean および Entity Bean（リモート呼び出し）のトレース取得ポイントの詳細

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB300	1	標準レベル	EJB コンテナーが EJB のビジネスメソッドを呼び出す直前	EJB の実装クラス名	メソッド名(引数の数)	-
0xB301	2	標準レベル	EJB のビジネスメソッドの呼び出し直後	EJB の実装クラス名	メソッド名(引数の数)	正常時 - 異常時 例外名※
0xBB00	7	標準レベル	RMI/IIOP リクエスト送信前	RMI/IIOP のオペレーション名	RMI/IIOP のオペレーション名	-
0xBB01	10	標準レベル	RMI/IIOP レスポンス受信後	RMI/IIOP のインターフェース名	RMI/IIOP のオペレーション名	-
0xBB02	8	標準レベル	RMI/IIOP リクエスト受信後	RMI/IIOP のインターフェース名	RMI/IIOP のオペレーション名	-
0xBB03	9	標準レベル	RMI/IIOP レスポンス送信前	RMI/IIOP のインターフェース名	RMI/IIOP のオペレーション名	-

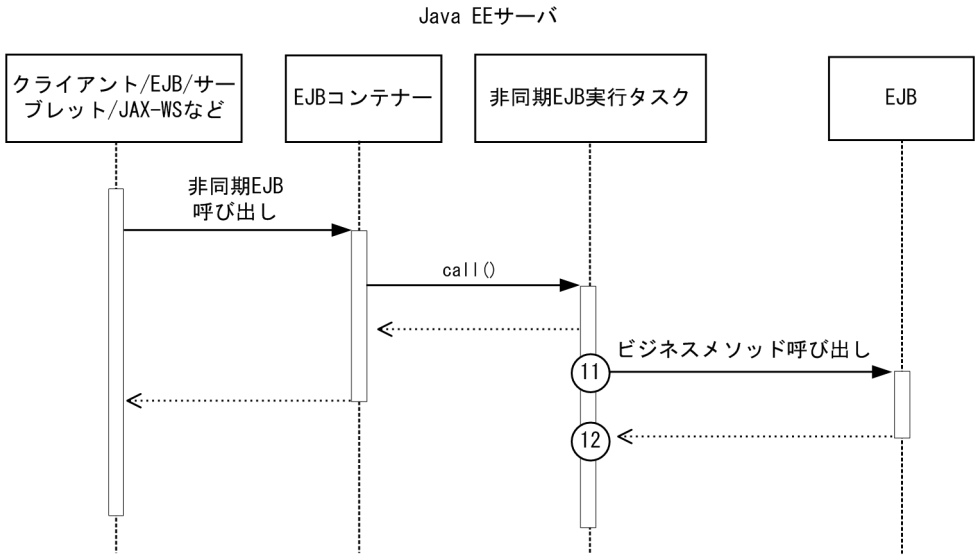
注※

トレース取得ポイントで取得した例外が java.lang.reflect.InvocationTargetException だった場合、java.lang.reflect.InvocationTargetException が保持している例外の例外名が出力されます。

非同期 EJB 呼び出しの場合

非同期 EJB 呼び出しでのトレースの取得ポイントを次に示します。

図 9-7　非同期 EJB 呼び出しのトレース取得ポイント



（凡例）

(n)：トレース取得ポイントを示します。

> **メモ**
>
> この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-9　非同期 EJB 呼び出しのトレース取得ポイントの詳細

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB306	11	標準レベル	非同期 EJB タスクが非同期 EJB のビジネスメソッドを呼び出す直前	EJB の実装クラス名	メソッド名(引数の数)	-
0xB307	12	標準レベル	非同期 EJB のビジネスメソッド呼び出し直後	EJB の実装クラス名	メソッド名(引数の数)	正常時 - 異常時 例外名※

注※

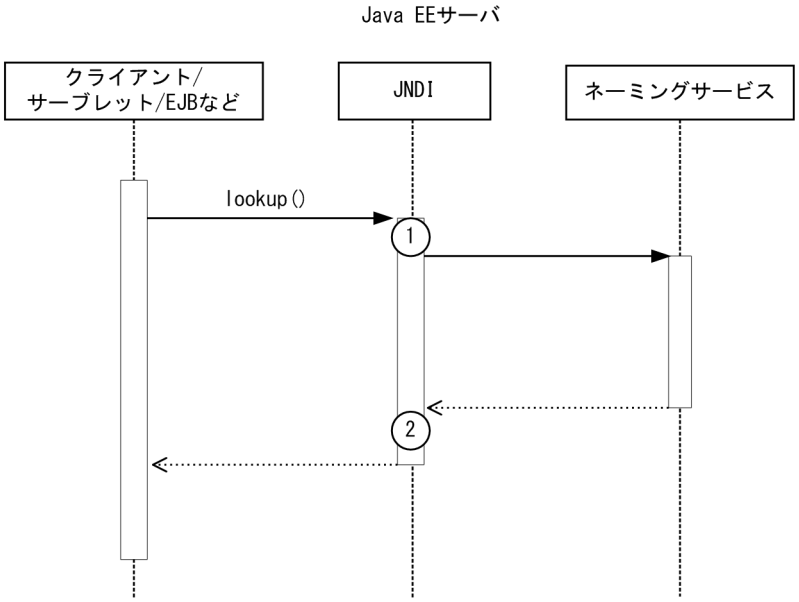
トレース取得ポイントで取得した例外が java.lang.reflect.InvocationTargetException だった場合、java.lang.reflect.InvocationTargetException が保持している例外の例外名が出力されます。

9.3.4 JNDI のトレース取得ポイント

JNDI でのトレース取得ポイントの詳細について説明します。

JNDI でのトレースの取得ポイントを次に示します。

図 9-8 JNDI のトレース取得ポイント



（凡例）

(n) ：トレース取得ポイントを示します。

メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-10　JNDI のトレース取得ポイントの詳細

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB400	1	標準レベル	javax.naming.Context.lookup()呼び出し直後（java: 名前空間検索時）	-	指定された名前	-
0xB401	2	標準レベル	javax.naming.Context.lookup()リターン直前（java: 名前空間検索時）	-	指定された名前	正常時 - 異常時 メッセージ：例外名

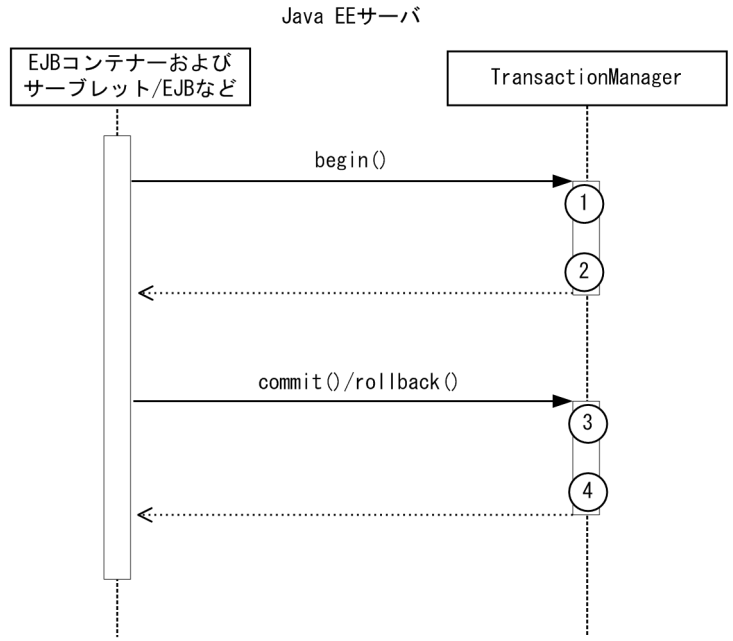
9.3.5　JTA のトレース取得ポイント

JTA でのトレース取得ポイントの詳細について説明します。

トランザクションを明示的に決着させる場合

トランザクションを明示的に決着させる場合の、JTA でのトレースの取得ポイントを次に示します。

図 9-9　JTA のトレース取得ポイント（トランザクションを明示的に決着させる場合）



（凡例）

(n)：トレース取得ポイントを示します。

> **メモ**
>
> この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-11　JTA のトレース取得ポイントの詳細（トランザクションを明示的に決着させる場合）

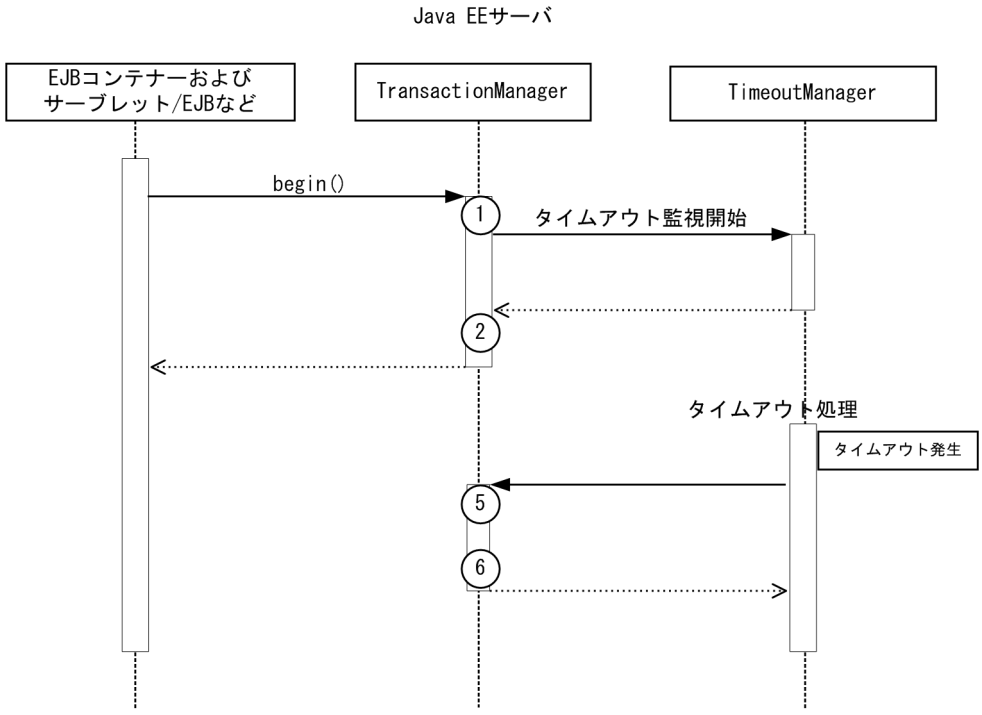
イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB500	1	標準レベル	トランザクション開始処理直前	-	-	-
0xB501	2	標準レベル	トランザクション開始処理直後	-	-	正常時 - 異常時 例外名
0xB502	3	標準レベル	トランザクションコミット処理直前	-	-	-

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB503	4	標準レベル	トランザクションコミット処理直後	-	-	正常時 - 異常時 例外名
0xB504	3	標準レベル	トランザクションロールバック処理直前	-	-	-
0xB505	4	標準レベル	トランザクションロールバック処理直後	-	-	正常時 - 異常時 例外名

トランザクションがタイムアウトする場合

トランザクションがタイムアウトする場合の、JTA でのトレースの取得ポイントを次に示します。

図 9-10　JTA のトレース取得ポイント（トランザクションがタイムアウトする場合）



（凡例）

(n)：トレース取得ポイントを示します。

> **メモ**
>
> この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-12　JTA のトレース取得ポイントの詳細（トランザクションがタイムアウトする場合）

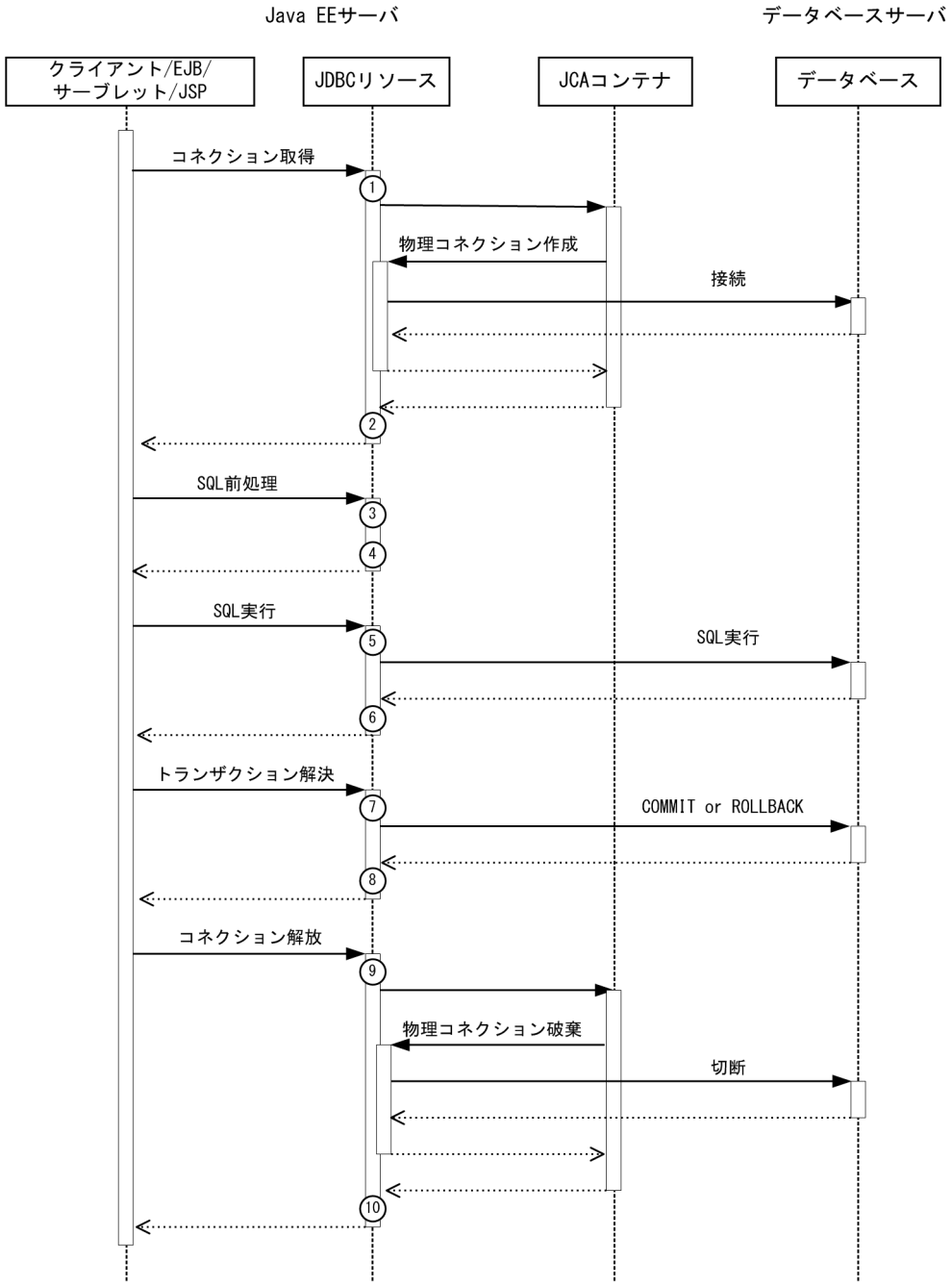
イベント ID	図中の番号	トレースレベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB500	1	標準レベル	トランザクション開始処理直前	-	-	-
0xB501	2	標準レベル	トランザクション開始処理直後	-	-	正常時 - 異常時 例外名
0xB506	5	標準レベル	トランザクションタイムアウト処理直前	-	-	-
0xB507	6	標準レベル	トランザクションタイムアウト処理直後	-	-	正常時 - 異常時 例外名

9.3.6　JDBC のトレース取得ポイント

JDBC でのトレース取得ポイントの詳細について説明します。

JDBC でのトレースの取得ポイントを次に示します。

図 9-11　JDBC のトレース取得ポイント



（凡例）

(n)：トレース取得ポイントを示します。

メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-13　JDBC のトレース取得ポイントの詳細

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB600	1	標準レベル	javax.sql.DataSource.getConnection()での、データベースコネクションの確立処理開始	-	-	-
0xB601	2	標準レベル	javax.sql.DataSource.getConnection()での、データベースコネクションの確立処理終了	-	-	正常時 コネクション ID 異常時 例外名
0xB602	1	標準レベル	javax.sql.DataSource.getConnection(String username, String password)での、データベースコネクションの確立処理開始	-	-	-
0xB603	2	標準レベル	javax.sql.DataSource.getConnection(String username, String password)での、データベースコネクションの確立処理終了	-	-	正常時 コネクション ID 異常時 例外名
0xB604	9	標準レベル	java.sql.Connection.close()での、Connection オブジェクトのデータベースと JDBC リソースの解除処理開始	-	-	コネクション ID
0xB605	10	標準レベル	java.sql.Connection.close()での、Connection オブジェクトのデータベースと JDBC リソースの解除処理終了	-	-	正常時 - 異常時 例外名
0xB606	7	標準レベル	java.sql.Connection.commit()処理開始	-	-	-
0xB607	8	標準レベル	java.sql.Connection.commit()処理終了	-	-	正常時 - 異常時 例外名

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB608	7	標準レベル	java.sql.Connection.rollback()処理開始	-	-	-
0xB609	8	標準レベル	java.sql.Connection.rollback()処理終了	-	-	正常時 - 異常時 例外名
0xB60A	7	標準レベル	java.sql.Connection.rollback(Savepoint savepoint)処理開始	-	-	-
0xB60B	8	標準レベル	java.sql.Connection.rollback(Savepoint savepoint)処理終了	-	-	正常時 - 異常時 例外名
0xB60C	3	標準レベル	Connection.prepareCall(String sql)処理開始	-	-	SQL 文
0xB60D	4	標準レベル	Connection.prepareCall(String sql)処理終了	-	-	正常時 - 異常時 例外名
0xB60E	3	標準レベル	Connection.prepareCall(String sql, int resultSetType, int resultSetConcurrency)処理開始	-	-	SQL 文
0xB60F	3	標準レベル	Connection.prepareCall(String sql, int resultSetType, int resultSetConcurrency)処理終了	-	-	正常時 - 異常時 例外名
0xB610	4	標準レベル	Connection.prepareCall(String sql, int resultSetType, int resultSetConcurrency, int resultSetHoldability)処理開始	-	-	SQL 文
0xB611	3	標準レベル	Connection.prepareCall(String sql, int resultSetType, int resultSetConcurrency, int resultSetHoldability)処理終了	-	-	正常時 - 異常時 例外名

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB612	3	標準レベル	Connection.prepareStatement(String sql)処理開始	-	-	SQL 文
0xB613	4	標準レベル	Connection.prepareStatement(String sql)処理終了	-	-	正常時 - 異常時 例外名
0xB614	3	標準レベル	Connection.prepareStatement(String sql, int autoGeneratedKeys)処理開始	-	-	SQL 文
0xB615	4	標準レベル	Connection.prepareStatement(String sql, int autoGeneratedKeys)処理終了	-	-	正常時 - 異常時 例外名
0xB616	3	標準レベル	Connection.prepareStatement(String sql, int[] columnIndexes)処理開始	-	-	SQL 文
0xB617	4	標準レベル	Connection.prepareStatement(String sql, int[] columnIndexes)処理終了	-	-	正常時 - 異常時 例外名
0xB618	3	標準レベル	Connection.prepareStatement(String sql, int resultSetType, int resultSetConcurrency)処理開始	-	-	SQL 文
0xB619	4	標準レベル	Connection.prepareStatement(String sql, int resultSetType, int resultSetConcurrency)処理終了	-	-	正常時 - 異常時 例外名
0xB61A	3	標準レベル	Connection.prepareStatement(String sql, int resultSetType, int resultSetConcurrency, int resultSetHoldability)処理開始	-	-	SQL 文

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB61B	4	標準レベル	Connection.prepareStatement(String sql, int resultSetType, int resultSetConcurrency, int resultSetHoldability)処理終了	-	-	正常時 - 異常時 例外名
0xB61C	3	標準レベル	Connection.prepareStatement(String sql, String[] columnNames)処理開始	-	-	SQL 文
0xB61D	4	標準レベル	Connection.prepareStatement(String sql, String[] columnNames)処理終了	-	-	正常時 - 異常時 例外名
0xB61E	5	標準レベル	Statement.execute(String sql)処理開始	-	-	SQL 文
0xB61F	6	標準レベル	Statement.execute(String sql)処理終了	-	-	正常時 - 異常時 例外名
0xB620	5	標準レベル	Statement.execute(String sql, int autoGeneratedKeys)処理開始	-	-	SQL 文
0xB621	6	標準レベル	Statement.execute(String sql, int autoGeneratedKeys)処理終了	-	-	正常時 - 異常時 例外名
0xB622	5	標準レベル	Statement.execute(String sql, int[] columnIndexes)処理開始	-	-	SQL 文
0xB623	6	標準レベル	Statement.execute(String sql, int[] columnIndexes)処理終了	-	-	正常時 - 異常時 例外名
0xB624	5	標準レベル	Statement.execute(String sql, String[] columnNames)処理開始	-	-	SQL 文

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB625	6	標準レベル	Statement.execute(String sql, String[] columnNames)処理終了	-	-	正常時 - 異常時 例外名
0xB626	5	標準レベル	Statement.executeBatch()処理開始	-	-	-
0xB627	6	標準レベル	Statement.executeBatch()処理終了	-	-	正常時 - 異常時 例外名
0xB628	5	標準レベル	Statement.executeQuery(String sql)処理開始	-	-	SQL 文
0xB629	6	標準レベル	Statement.executeQuery(String sql)処理終了	-	-	正常時 - 異常時 例外名
0xB62A	5	標準レベル	Statement.executeUpdate(String sql)処理開始	-	-	SQL 文
0xB62B	6	標準レベル	Statement.executeUpdate(String sql)処理終了	-	-	正常時 - 異常時 例外名
0xB62C	5	標準レベル	Statement.executeUpdate(String sql, int autoGeneratedKeys)処理開始	-	-	SQL 文
0xB62D	6	標準レベル	Statement.executeUpdate(String sql, int autoGeneratedKeys)処理終了	-	-	正常時 - 異常時 例外名
0xB62E	5	標準レベル	Statement.executeUpdate(String sql, int[] columnIndexes)処理開始	-	-	SQL 文

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB62F	6	標準レベル	Statement.executeUpdate(String sql, int[] columnIndexes)処理終了	-	-	正常時 - 異常時 例外名
0xB630	5	標準レベル	Statement.executeUpdate(String sql, String[] columnNames)処理開始	-	-	SQL 文
0xB631	6	標準レベル	Statement.executeUpdate(String sql, String[] columnNames)処理終了	-	-	正常時 - 異常時 例外名
0xB632	5	標準レベル	PreparedStatement.execute()処理開始	-	-	-
0xB633	6	標準レベル	PreparedStatement.execute()処理終了	-	-	正常時 - 異常時 例外名
0xB634	5	標準レベル	PreparedStatement.execute(String sql)処理開始	-	-	SQL 文
0xB635	6	標準レベル	PreparedStatement.execute(String sql)処理終了	-	-	正常時 - 異常時 例外名
0xB636	5	標準レベル	PreparedStatement.execute(String sql, int autoGeneratedKeys)処理開始	-	-	SQL 文
0xB637	6	標準レベル	PreparedStatement.execute(String sql, int autoGeneratedKeys)処理終了	-	-	正常時 - 異常時 例外名
0xB638	5	標準レベル	PreparedStatement.execute(String sql, int[] columnIndexes)処理開始	-	-	SQL 文

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB639	6	標準レベル	PreparedStatement.execute(String sql, int[] columnIndexes)処理終了	-	-	正常時 - 異常時 例外名
0xB63A	5	標準レベル	PreparedStatement.execute(String sql, String[] columnNames)処理開始	-	-	SQL 文
0xB63B	6	標準レベル	PreparedStatement.execute(String sql, String[] columnNames)処理終了	-	-	正常時 - 異常時 例外名
0xB63C	5	標準レベル	PreparedStatement.executeBatch()処理開始	-	-	-
0xB63D	6	標準レベル	PreparedStatement.executeBatch()処理終了	-	-	正常時 - 異常時 例外名
0xB63E	5	標準レベル	PreparedStatement.executeQuery()処理開始	-	-	-
0xB63F	6	標準レベル	PreparedStatement.executeQuery()処理終了	-	-	正常時 - 異常時 例外名
0xB640	5	標準レベル	PreparedStatement.executeQuery(String sql)処理開始	-	-	SQL 文
0xB641	6	標準レベル	PreparedStatement.executeQuery(String sql)処理終了	-	-	正常時 - 異常時 例外名
0xB642	5	標準レベル	PreparedStatement.executeUpdate()処理開始	-	-	-
0xB643	6	標準レベル	PreparedStatement.executeUpdate()処理終了	-	-	正常時 - 異常時 例外名

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB644	5	標準レベル	PreparedStatement.executeUpdate(String sql)処理開始	-	-	SQL 文
0xB645	6	標準レベル	PreparedStatement.executeUpdate(String sql)処理終了	-	-	正常時 - 異常時 例外名
0xB646	5	標準レベル	PreparedStatement.executeUpdate(String sql, int autoGeneratedKeys)処理開始	-	-	SQL 文
0xB647	6	標準レベル	PreparedStatement.executeUpdate(String sql, int autoGeneratedKeys)処理終了	-	-	正常時 - 異常時 例外名
0xB648	5	標準レベル	PreparedStatement.executeUpdate(String sql, int[] columnIndexes)処理開始	-	-	SQL 文
0xB649	6	標準レベル	PreparedStatement.executeUpdate(String sql, int[] columnIndexes)処理終了	-	-	正常時 - 異常時 例外名
0xB64A	5	標準レベル	PreparedStatement.executeUpdate(String sql, String[] columnNames)処理開始	-	-	SQL 文
0xB64B	6	標準レベル	PreparedStatement.executeUpdate(String sql, String[] columnNames)処理終了	-	-	正常時 - 異常時 例外名
0xB64C	5	標準レベル	CallableStatement.execute()処理開始	-	-	-
0xB64D	6	標準レベル	CallableStatement.execute()処理終了	-	-	正常時 - 異常時 例外名

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB64E	5	標準レベル	CallableStatement.execute(String sql)処理開始	-	-	SQL 文
0xB64F	6	標準レベル	CallableStatement.execute(String sql)処理終了	-	-	正常時 - 異常時 例外名
0xB650	5	標準レベル	CallableStatement.execute(String sql, int autoGeneratedKeys)処理開始	-	-	SQL 文
0xB651	6	標準レベル	CallableStatement.execute(String sql, int autoGeneratedKeys)処理終了	-	-	正常時 - 異常時 例外名
0xB652	5	標準レベル	CallableStatement.execute(String sql, int[] columnIndexes)処理開始	-	-	SQL 文
0xB653	6	標準レベル	CallableStatement.execute(String sql, int[] columnIndexes)処理終了	-	-	正常時 - 異常時 例外名
0xB654	5	標準レベル	CallableStatement.execute(String sql, String[] columnNames)処理開始	-	-	SQL 文
0xB655	6	標準レベル	CallableStatement.execute(String sql, String[] columnNames)処理終了	-	-	正常時 - 異常時 例外名
0xB656	5	標準レベル	CallableStatement.executeBatch()処理開始	-	-	-
0xB657	6	標準レベル	CallableStatement.executeBatch()処理終了	-	-	正常時 - 異常時 例外名

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB658	5	標準レベル	CallableStatement.executeQuery()処理開始	-	-	-
0xB659	6	標準レベル	CallableStatement.executeQuery()処理終了	-	-	正常時 - 異常時 例外名
0xB65A	5	標準レベル	CallableStatement.executeQuery(String sql)処理開始	-	-	SQL 文
0xB65B	6	標準レベル	CallableStatement.executeQuery(String sql)処理終了	-	-	正常時 - 異常時 例外名
0xB65C	5	標準レベル	CallableStatement.executeUpdate()処理開始	-	-	-
0xB65D	6	標準レベル	CallableStatement.executeUpdate()処理終了	-	-	正常時 - 異常時 例外名
0xB65E	5	標準レベル	CallableStatement.executeUpdate(String sql)処理開始	-	-	SQL 文
0xB65F	6	標準レベル	CallableStatement.executeUpdate(String sql)処理終了	-	-	正常時 - 異常時 例外名
0xB660	5	標準レベル	CallableStatement.executeUpdate(String sql, int autoGeneratedKeys)処理開始	-	-	SQL 文
0xB661	6	標準レベル	CallableStatement.executeUpdate(String sql, int autoGeneratedKeys)処理終了	-	-	正常時 - 異常時 例外名
0xB662	5	標準レベル	CallableStatement.executeUpdate(String sql, int[] columnIndexes)処理開始	-	-	SQL 文

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB663	6	標準レベル	CallableStatement.executeUpdate(String sql, int[] columnIndexes)処理終了	-	-	正常時 - 異常時 例外名
0xB664	5	標準レベル	CallableStatement.executeUpdate(String sql, String[] columnNames)処理開始	-	-	SQL 文
0xB665	6	標準レベル	CallableStatement.executeUpdate(String sql, String[] columnNames)処理終了	-	-	正常時 - 異常時 例外名

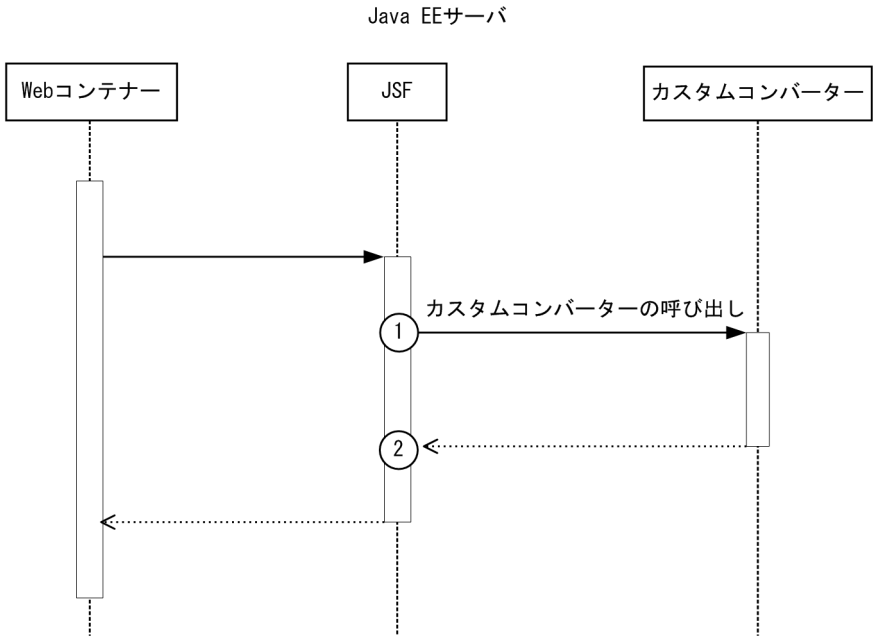
9.3.7 JSF のトレース取得ポイント

JSF でのトレース取得ポイントの詳細について説明します。

カスタムコンバーターが呼び出される場合

カスタムコンバーターが呼び出される場合の、JSF でのトレースの取得ポイントを次に示します。

図 9-12　JSF のトレース取得ポイント（カスタムコンバーターが呼び出される場合）



（凡例）

(n)：トレース取得ポイントを示します。

メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

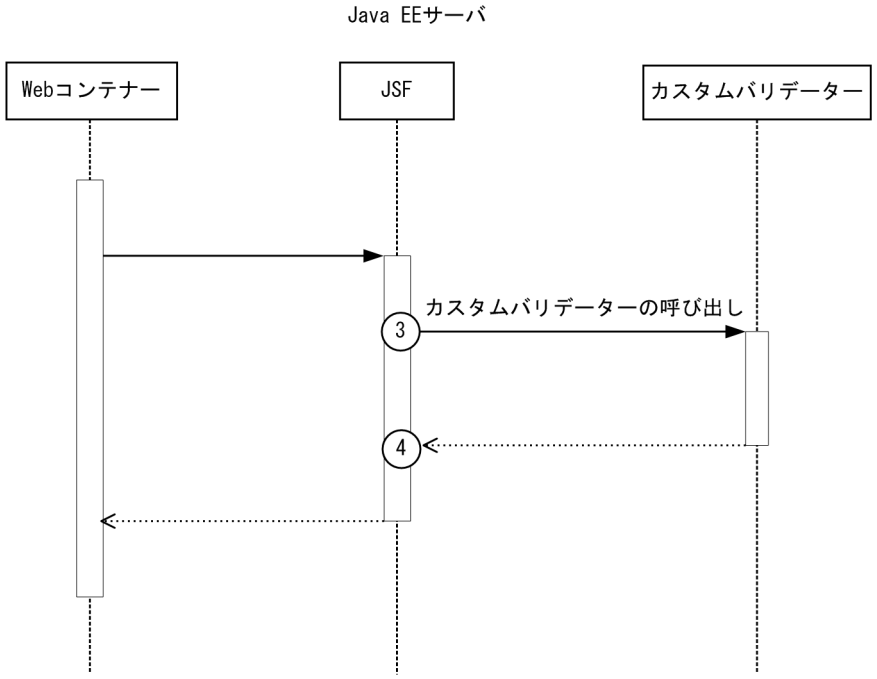
表 9-14　JSF のトレース取得ポイントの詳細（カスタムコンバーターが呼び出される場合）

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB700	1	標準レベル	カスタムコンバーターが呼び出される直前	クライアント ID	カスタムコンバーターのクラス名とメソッド名です。	-
0xB701	2	標準レベル	カスタムコンバーターの処理が終了した直後	クライアント ID	カスタムコンバーターのクラス名とメソッド名です。	正常時 - 異常時 例外名

カスタムバリデーターが呼び出される場合

カスタムバリデーターが呼び出される場合の、JSF でのトレースの取得ポイントを次に示します。

図 9-13　JSF のトレース取得ポイント（カスタムバリデーターが呼び出される場合）



（凡例）

(n)：トレース取得ポイントを示します。

> **メモ**
>
> この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-15　JSF のトレース取得ポイントの詳細（カスタムバリデーターが呼び出される場合）

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB702	3	標準レベル	カスタムバリデーターが呼び出される直前	クライアント ID	カスタムバリデーターのクラス名です。MethodExpression を利用してカスタムバリデーターを呼び出す場合※は	-

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
					MethodExpression です。	
0xB703	4	標準レベル	カスタムバリデーターの処理が終了した直後	クライアント ID	カスタムバリデーターのクラス名です。MethodExpression を利用してカスタムバリデーターを呼び出す場合※は MethodExpression です。	正常時 - 異常時 例外名

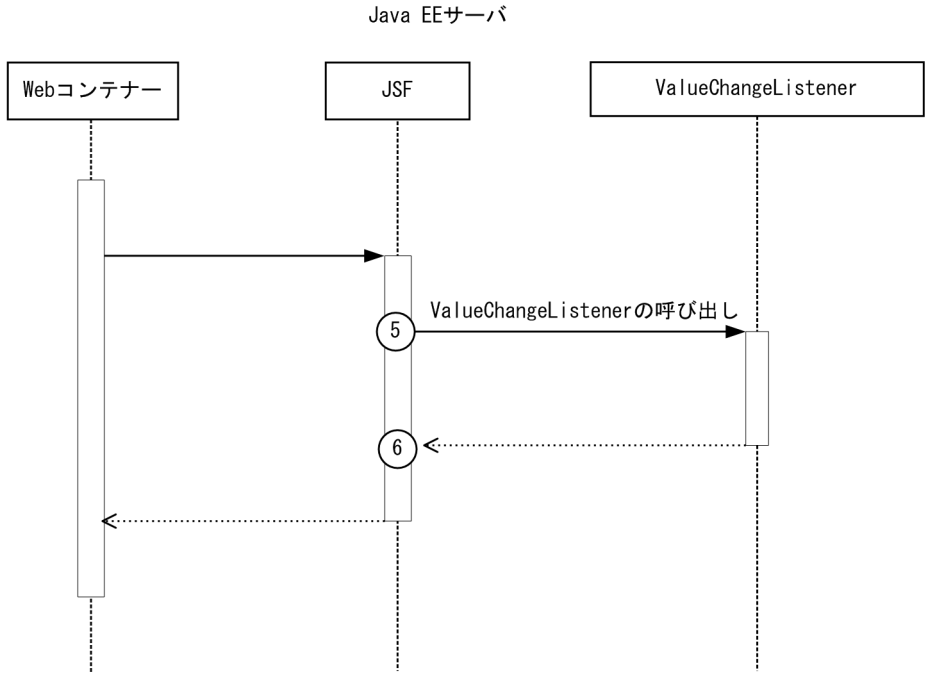
注※

引数を複数記述した MethodExpression を使用した場合、性能解析トレースファイル（CSV 形式）の列がずれることがあります。

ValueChangeListener が呼び出される場合

ValueChangeListener が呼び出される場合の、JSF でのトレースの取得ポイントを次に示します。

図 9-14　JSF のトレース取得ポイント（ValueChangeListener が呼び出される場合）



（凡例）

(n)：トレース取得ポイントを示します。

メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

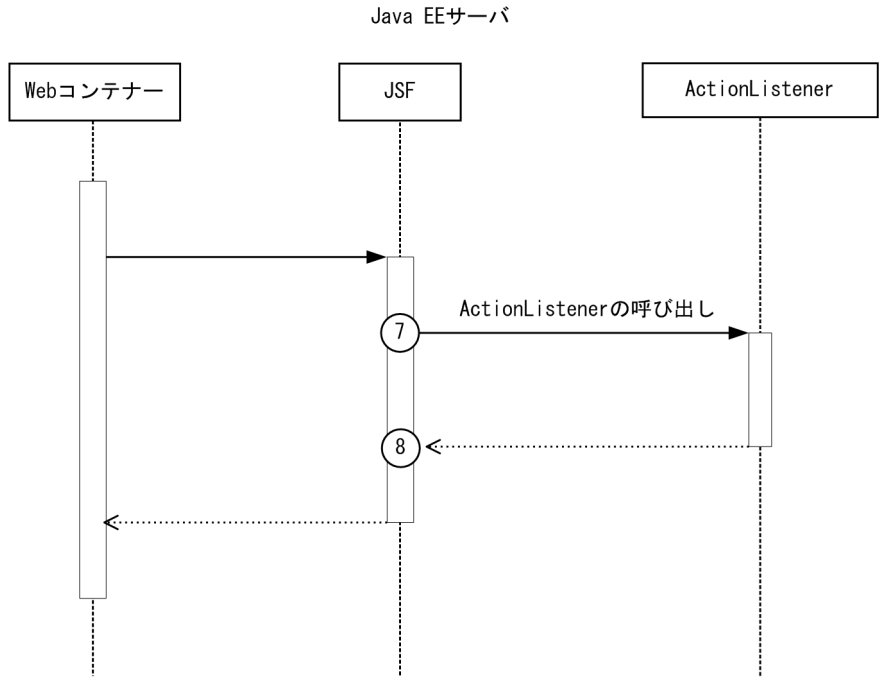
表 9-16　JSF のトレース取得ポイントの詳細（ValueChangeListener が呼び出される場合）

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB704	5	標準レベル	ValueChangeListener が呼び出される直前	クライアント ID	ValueChangeListener のクラス名です。 MethodExpression を利用して ValueChangeListener を呼び出す場合、MethodExpression です。	-
0xB705	6	標準レベル	ValueChangeListener の処理が終了した直後	クライアント ID	正常時 MethodExpression で ValueChangeListener が呼び出されて正常終了した場合、1 つの引数を持つメソッドが呼び出された時は`one argument` を出力します。引数のないメソッドが呼ばれた時は`no argument` を出力します。 異常時 -	正常時 - 異常時 例外名

ActionListener が呼び出される場合

ActionListener が呼び出される場合の、JSF でのトレースの取得ポイントを次に示します。

図 9-15　JSF のトレース取得ポイント（ActionListener が呼び出される場合）



（凡例）

(n)：トレース取得ポイントを示します。

メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベ ActionListener ル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-17　JSF のトレース取得ポイントの詳細（が呼び出される場合）

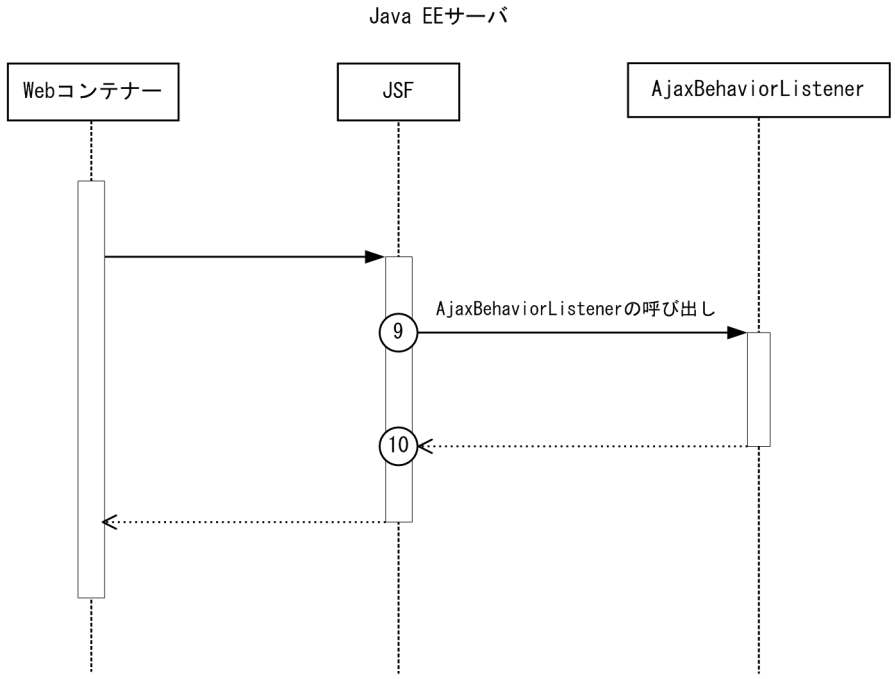
イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB706	7	標準レベル	ActionListener が呼び出される直前	クライアント ID	ActionListener のクラス名です。 MethodExpression を利用して ActionListener を呼び出す場合、MethodExpression です。	-

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB707	8	標準レベル	ActionListener の処理が終了した直後	クライアント ID	正常時 MethodExpression で ActionListener が呼び出されて正常終了した場合、1 つの引数を持つメソッドが呼び出された時は one argument を出力します。引数のないメソッドが呼ばれた時はno argument を出力します。 異常時 -	正常時 - 異常時 例外名

AjaxBehaviorListener が呼び出される場合

AjaxBehaviorListener が呼び出される場合の、JSF でのトレースの取得ポイントを次に示します。

図 9-16　JSF のトレース取得ポイント（AjaxBehaviorListener が呼び出される場合）



（凡例）

(n)：トレース取得ポイントを示します。

メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

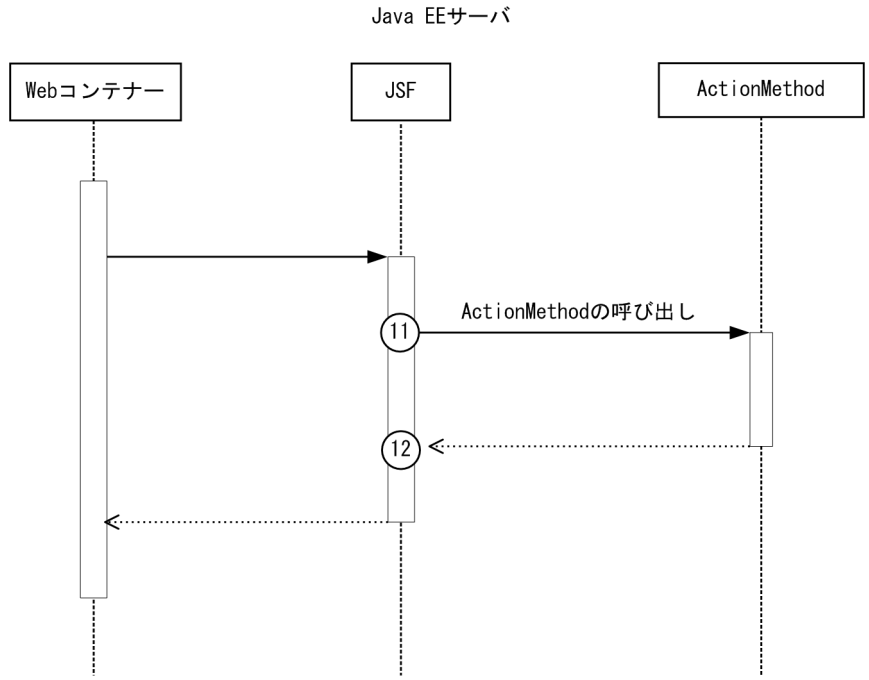
表 9-18　JSF のトレース取得ポイントの詳細（AjaxBehaviorListener が呼び出される場合）

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB708	9	PRF トレース取得レベル	AjaxBehaviorListener が呼び出される直前	クライアント ID	AjaxBehaviorListener のクラス名です。 MethodExpression を利用して AjaxBehaviorListener を呼び出す場合、MethodExpression です。	-
0xB709	10	標準レベル	AjaxBehaviorListener の処理が終了した直後	クライアント ID	正常時 MethodExpression で AjaxBehaviorListener が呼び出されて正常終了した場合、1 つの引数を持つメソッドが呼び出された時は`one argument` を出力します。引数のないメソッドが呼ばれた時は`no argument` を出力します。 異常時 -	正常時 - 異常時 例外名

ActionMethod が呼び出される場合

ActionMethod が呼び出される場合の、JSF でのトレースの取得ポイントを次に示します。

図 9-17　JSF のトレース取得ポイント（ActionMethod が呼び出される場合）



（凡例）

(n)：トレース取得ポイントを示します。

> **メモ**
>
> この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

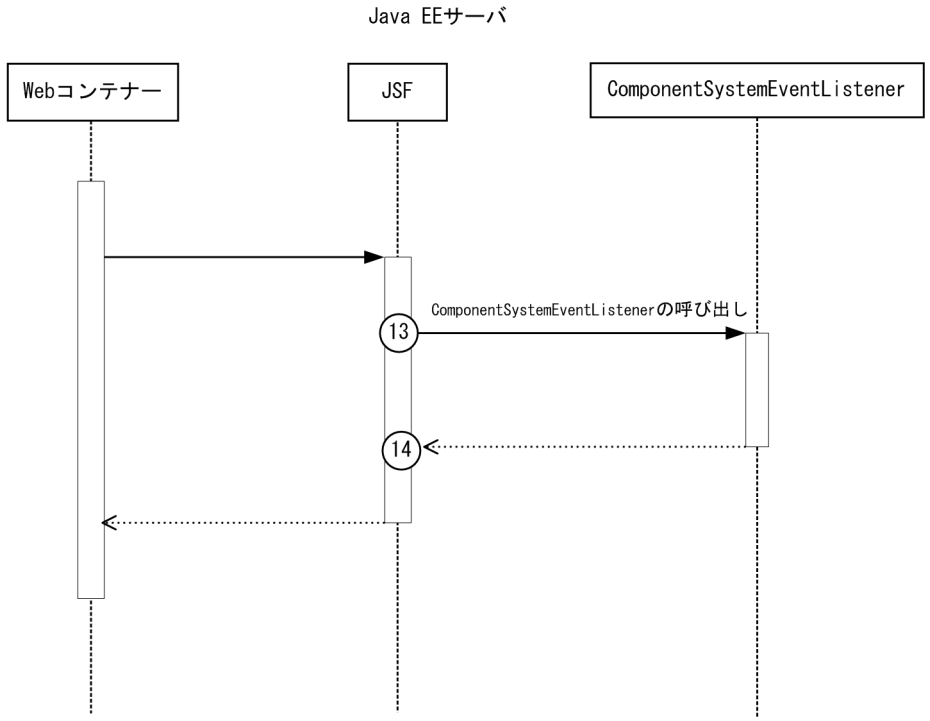
表 9-19　JSF のトレース取得ポイントの詳細（ActionMethod が呼び出される場合）

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB70A	11	標準レベル	ActionMethod が呼び出される直前	クライアント ID	-	-
0xB70B	12	標準レベル	ActionMethod の処理が終了した直後	クライアント ID	-	正常時 - 異常時 例外名

ComponentSystemEventListener が呼び出される場合

ComponentSystemEventListener が呼び出される場合の、JSF でのトレースの取得ポイントを次に示します。

図 9-18　JSF のトレース取得ポイント（ComponentSystemEventListener が呼び出される場合）



（凡例）

(n)：トレース取得ポイントを示します。

> **メモ**
>
> この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-20　JSF のトレース取得ポイントの詳細（ComponentSystemEventListener が呼び出される場合）

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB70C	13	標準レベル	ComponentSystemEventListener が呼び出される直前	クライアント ID	ComponentSystemEventListener のクラス名です。 MethodExpression を利用して ComponentSystemEventListener を呼び出す場合、	-

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
					MethodExpression です。	
0xB70D	14	標準レベル	ComponentSystemEventListener の処理が終了した直後	クライアント ID	正常時 MethodExpression で ComponentSystemEventListener が呼び出されて正常終了した場合、1 つの引数を持つメソッドが呼び出された時は「one argument」を出力します。引数のないメソッドが呼ばれた時は「no argument」を出力します。 異常時 -	正常時 - 異常時 例外名

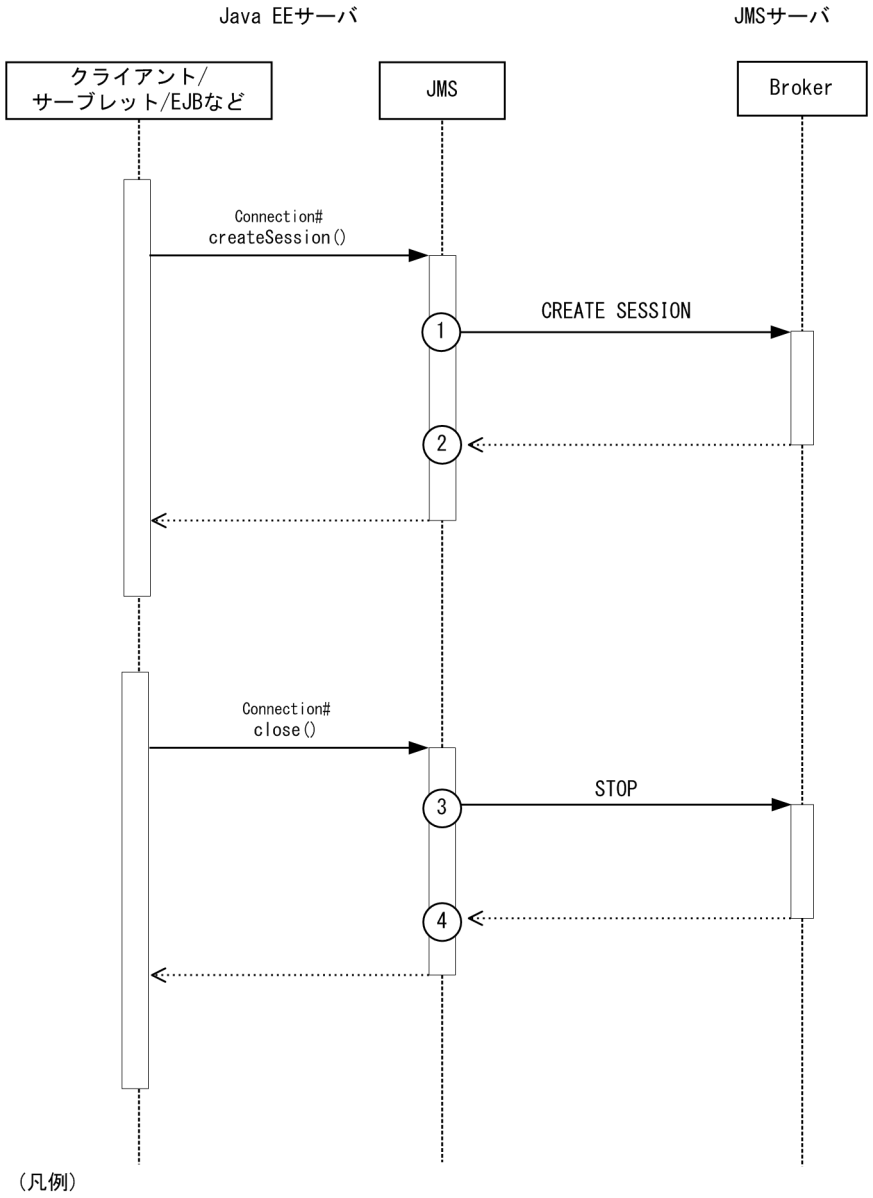
9.3.8 JMS のトレース取得ポイント

JMS でのトレース取得ポイントの詳細について説明します。

Connection オブジェクト操作の場合

Connection オブジェクト操作の場合の、JMS でのトレースの取得ポイントを次に示します。

図 9-19　JMS のトレース取得ポイント（Connection オブジェクト操作の場合）



メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

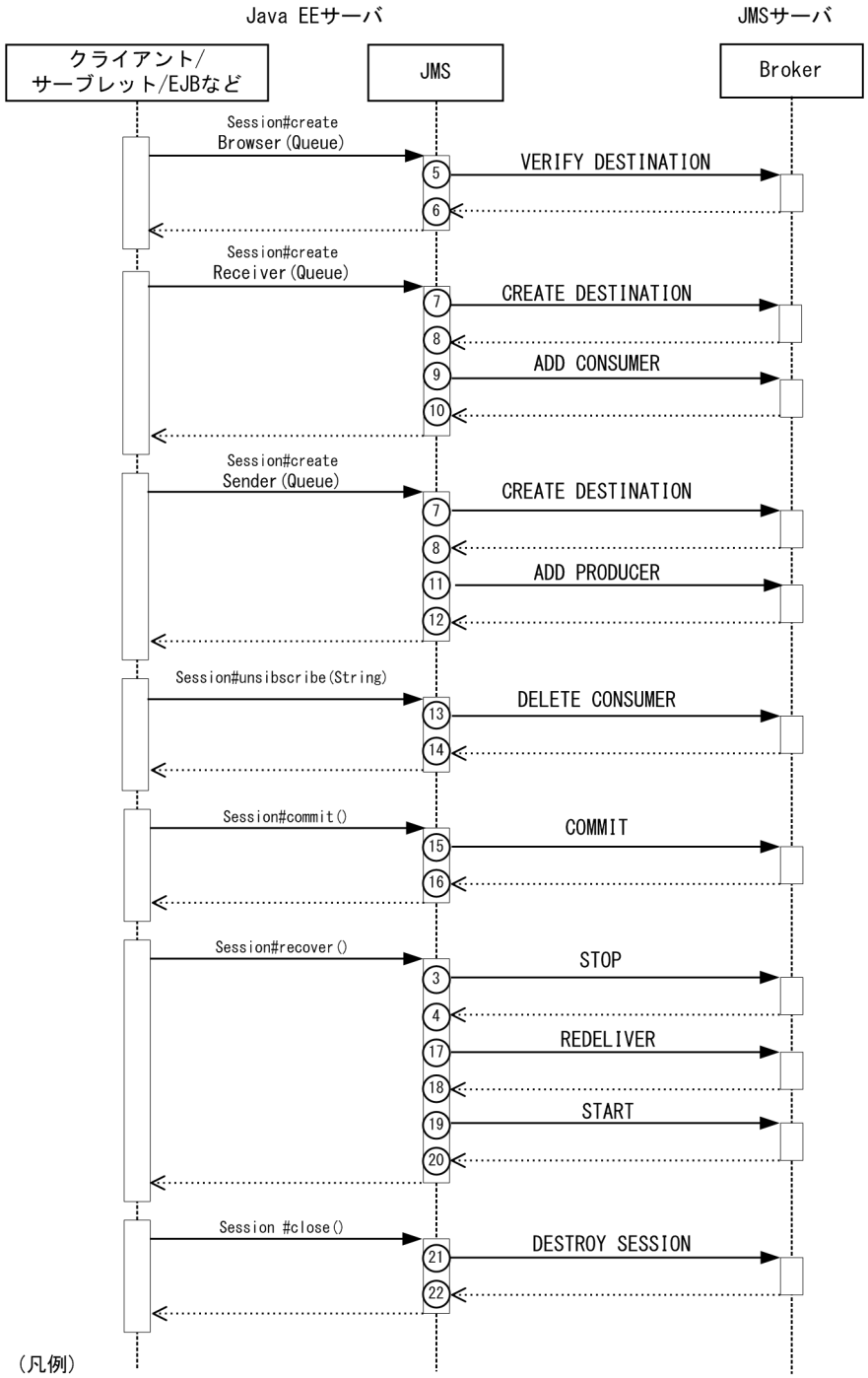
表 9-21　JMS のトレース取得ポイントの詳細（Connection オブジェクト操作の場合）

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB800	1	標準レベル	Broker へ CREATE SESSION を送信する直前	-	-	-
0xB801	2	標準レベル	Broker から CREATE SESSION の応答を受信した直後	-	-	-
0xB812	3	標準レベル	Broker へ STOP を送信する直前	-	-	-
0xB813	4	標準レベル	Broker から STOP の応答を受信した直後	-	-	-

Session オブジェクト操作の場合

Session オブジェクト操作の場合の、JMS でのトレースの取得ポイントを次に示します。

図 9-20　JMS のトレース取得ポイント（Session オブジェクト操作の場合）



メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-22　JMS のトレース取得ポイントの詳細（Session オブジェクト操作の場合）

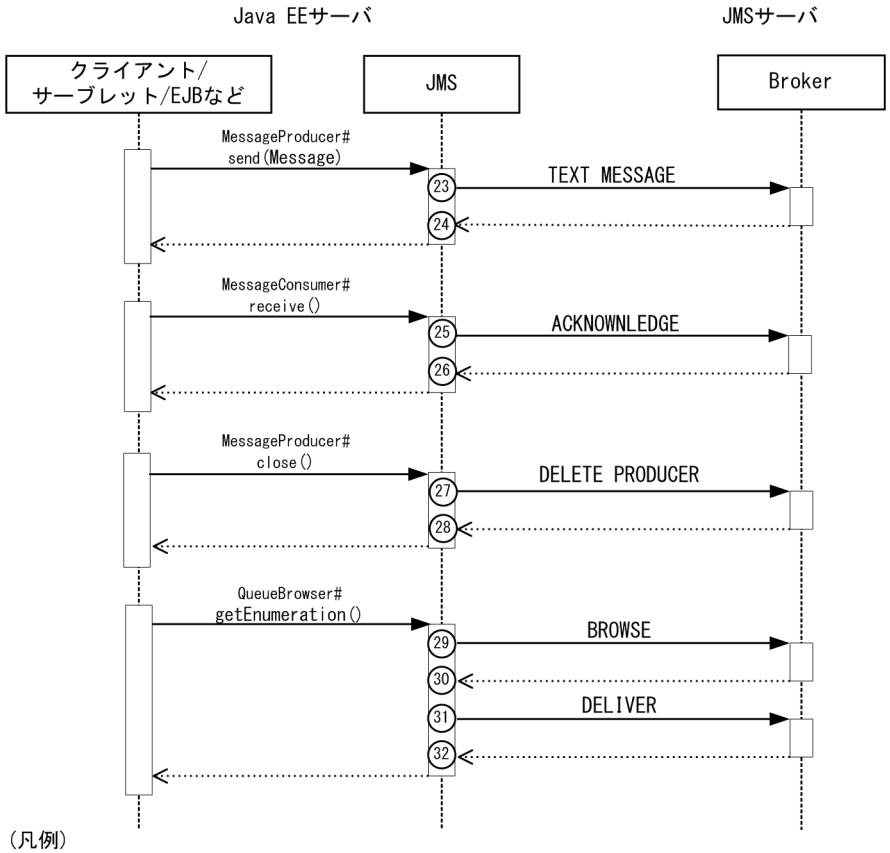
イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB802	21	標準レベル	Broker へ DESTROY SESSION を送信する直前	-	-	-
0xB803	22	標準レベル	Broker から DESTROY SESSION の応答を受信した直後	-	-	-
0xB804	7	標準レベル	Broker へ CREATE DESTINATION を送信する直前	-	-	-
0xB805	8	標準レベル	Broker から CREATE DESTINATION の応答を受信した直後	-	-	-
0xB806	5	標準レベル	Broker へ VERIFY DESTINATION を送信する直前	-	-	-
0xB807	6	標準レベル	Broker から VERIFY DESTINATION の応答を受信した直後	-	-	-
0xB808	9	標準レベル	Broker へ ADD CONSUMER を送信する直前	-	-	-
0xB809	10	標準レベル	Broker から ADD CONSUMER の応答を受信した直後	-	-	-
0xB80A	13	標準レベル	Broker へ DELETE CONSUMER を送信する直前	-	-	-
0xB80B	14	標準レベル	Broker から DELETE CONSUMER の応答を受信した直後	-	-	-
0xB80C	11	標準レベル	Broker へ ADD PRODUCER を送信する直前	-	-	-
0xB80D	12	標準レベル	Broker から ADD PRODUCER の応答を受信した直後	-	-	-
0xB810	19	標準レベル	Broker へ START を送信する直前	-	-	-

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB811	20	標準レベル	Broker から START の応答を受信した直後	-	-	-
0xB812	3	標準レベル	Broker へ STOP を送信する直前	-	-	-
0xB813	4	標準レベル	Broker から STOP の応答を受信した直後	-	-	-
0xB814	15	標準レベル	Broker へ COMMIT または ROLLBACK を送信する直前	-	-	-
0xB815	16	標準レベル	Broker から COMMIT または ROLLBACK の応答を受信した直後	-	-	-
0xB81E	17	標準レベル	Broker へ REDELIVER を送信する直前	-	-	-
0xB81F	18	標準レベル	Broker から REDELIVER の応答を受信した直後	-	-	-

その他のオブジェクト操作の場合

その他のオブジェクト操作の場合の、JMS でのトレースの取得ポイントを次に示します。

図 9-21　JMS のトレース取得ポイント（その他のオブジェクト操作の場合）



（凡例）

(n)：トレース取得ポイントを示します。

メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-23　JMS のトレース取得ポイントの詳細（その他のオブジェクト操作の場合）

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB80E	27	標準レベル	Broker へ DELETE PRODUCER を送信する直前	-	-	-
0xB80F	28	標準レベル	Broker から DELETE PRODUCER の応答を受信した直後	-	-	-
0xB816	23	標準レベル	Broker へ TEXT MESSAGE、MAP	-	-	-

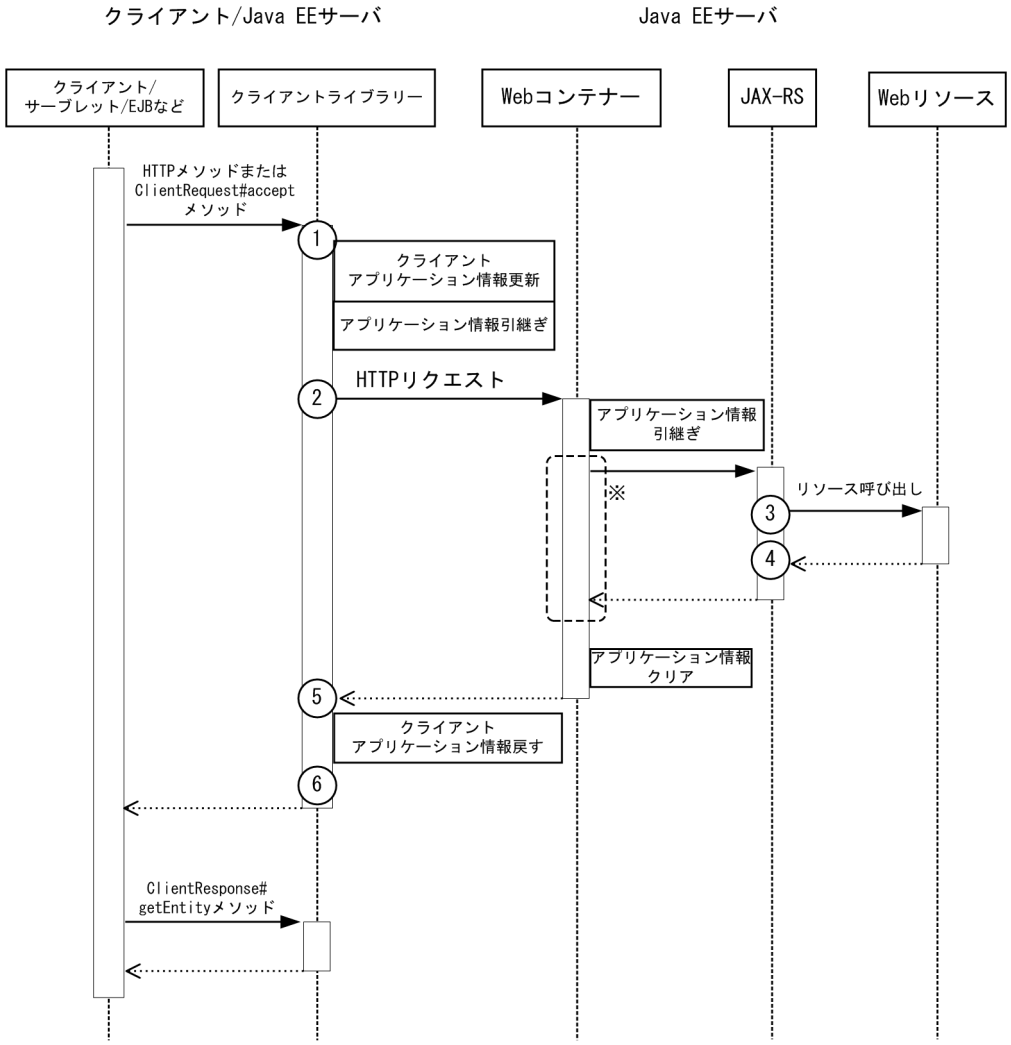
イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
			MESSAGE、BYTE MESSAGE、または OBJECT MESSAGE を送信する直前			
0xB817	24	標準レベル	Broker から TEXT MESSAGE、MAP MESSAGE、BYTE MESSAGE、または OBJECT MESSAGE の応答を受信した直後	-	-	-
0xB818	25	標準レベル	Broker へ ACKNOWLEDGE を送信する直前	-	-	-
0xB819	26	標準レベル	Broker から ACKNOWLEDGE の応答を受信した直後	-	-	-
0xB81A	29	標準レベル	Broker へ BROWSE を送信する直前	-	-	-
0xB81B	30	標準レベル	Broker から BROWSE の応答を受信した直後	-	-	-
0xB81C	31	標準レベル	Broker へ DELIVER を送信する直前	-	-	-
0xB81D	32	標準レベル	Broker から DELIVER の応答を受信した直後	-	-	-

9.3.9 JAX-RS のトレース取得ポイント

JAX-RS でのトレース取得ポイントの詳細について説明します。

JAX-RS でのトレースの取得ポイントを次に示します。

図 9-22　JAX-RS のトレース取得ポイント



（凡例）

(n)：トレース取得ポイントを示します。

メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

注※

点線で囲まれた部分は、同期処理の場合の Web コンテナーのトレース取得ポイントと同じです。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-24　JAX-RS のトレース取得ポイントの詳細

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xB900	1	標準レベル	HTTP メッセージを送信するクライアントライブラリーのメソッド入口	クラス名	メソッド名	-
0xB901	6	標準レベル	HTTP メッセージを送信するクライアントライブラリーのメソッド出口	クラス名	メソッド名	正常時 - 異常時 例外名
0xB902	2	標準レベル	クライアントライブラリーの HTTP メッセージ送信前	クラス名	メソッド名	エンドポイント URI
0xB903	5	標準レベル	クライアントライブラリーの HTTP メッセージ受信後	クラス名	メソッド名	正常時 - 異常時 例外名
0xB904	3	標準レベル	リソース呼び出し前	クラス名	メソッド名	呼び出し種別 次の中からどれかを出力します。 • ObjectOutInvoker • ResponseOutInvoker • TypeOutInvoker • VoidOutInvoker • VoidToVoidDispatcher
0xB905	4	標準レベル	リソース呼び出し後	クラス名	メソッド名	正常時 - 異常時 例外名

関連項目

- 9.3.2　Web コンテナーのトレース取得ポイント

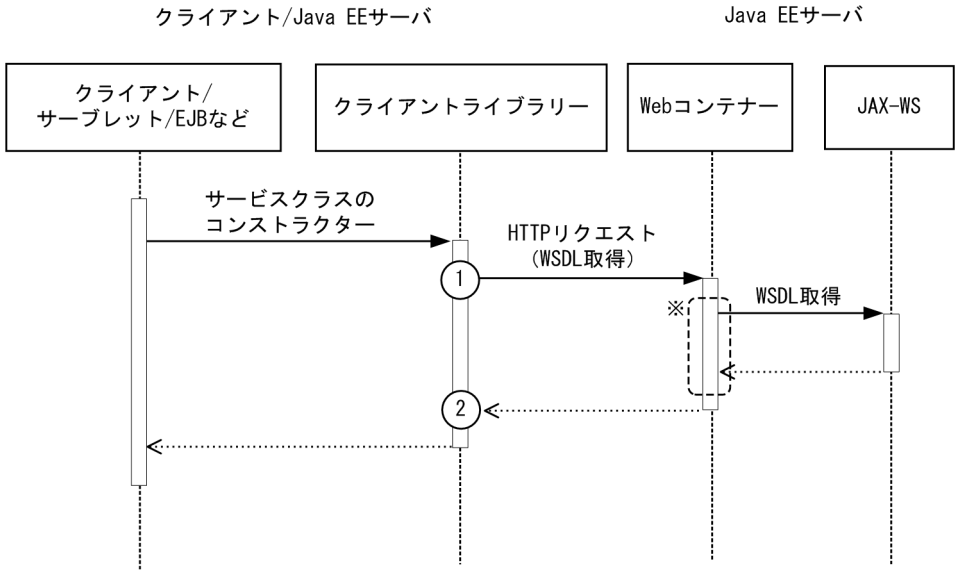
9.3.10　JAX-WS のトレース取得ポイント

JAX-WS でのトレース取得ポイントの詳細について説明します。

リモート WSDL 取得の場合

リモート WSDL 取得の場合の、JAX-WS でのトレースの取得ポイントを次に示します。

図 9-23　JAX-WS のトレース取得ポイント（リモート WSDL 取得の場合）



（凡例）

(n)：トレース取得ポイントを示します。

> **メモ**
>
> この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

注※

点線で囲まれた部分は、同期処理の場合の Web コンテナーのトレース取得ポイントと同じです。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-25　JAX-WS のトレース取得ポイントの詳細（リモート WSDL 取得の場合）

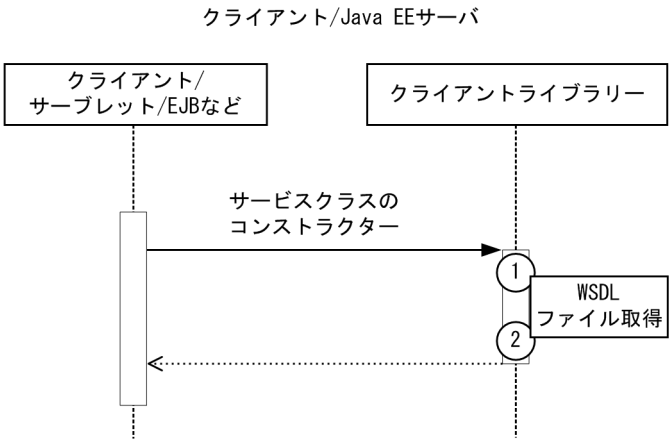
イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBA0A	1	標準レベル	WSDL 取得実行前	クラス名	メソッド名(引数の数)	エンドポイント URI

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBA0B	2	標準レベル	WSDL 取得実行後	クラス名	メソッド名 (引数の数)	正常時 - 異常時 例外名

ローカル WSDL 取得の場合

ローカル WSDL 取得の場合の、JAX-WS でのトレースの取得ポイントを次に示します。

図 9-24　JAX-WS のトレース取得ポイント（ローカル WSDL 取得の場合）



（凡例）

(n)：トレース取得ポイントを示します。

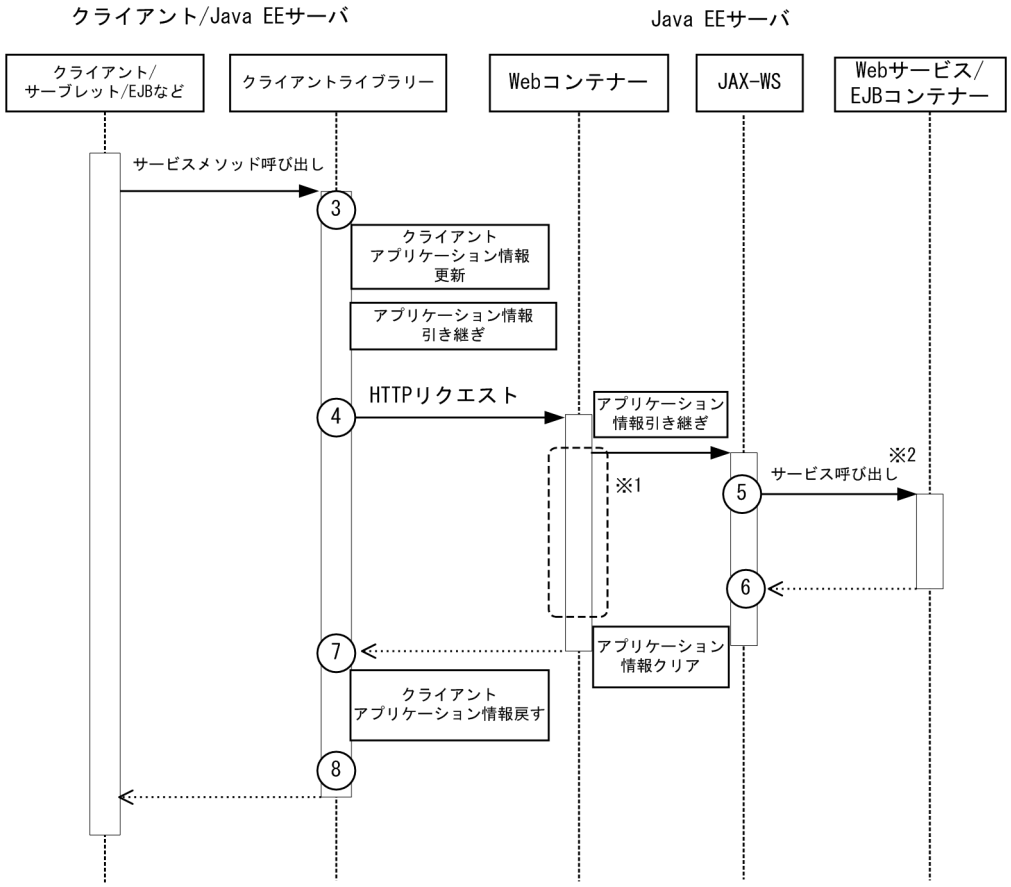
> **メモ**
>
> この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報については、リモート WSDL 取得の場合と同じです。

request-response オペレーション（同期）の場合

request-response オペレーション（同期）の場合の、JAX-WS でのトレースの取得ポイントを次に示します。

図 9-25　JAX-WS のトレース取得ポイント（request-response オペレーション（同期）の場合）



（凡例）

(n)：トレース取得ポイントを示します。

メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

注※1

点線で囲まれた部分は、同期処理の場合の Web コンテナーのトレース取得ポイントと同じです。

注※2

EJB コンテナーは、EJB を Web サービスとして実装している場合に呼び出されます。

EJB コンテナー以降の取得個所は、Session Bean および Entity Bean のローカル呼び出しのときと同じです。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-26　JAX-WS のトレース取得ポイントの詳細（request-response オペレーション（同期）の場合）

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBA00	3	標準レベル	SOAP クライアントライブラリーのサービスメソッド入口（スタブベース）	クラス名	メソッド名（引数の数）	-
0xBA01	8	標準レベル	SOAP クライアントライブラリーのサービスメソッド出口（スタブベース）	クラス名	メソッド名（引数の数）	正常時 　- 異常時 　例外名
0xBA02	3	標準レベル	SOAP クライアントライブラリーのサービスメソッド入口（ディスパッチベース）	クラス名	メソッド名（引数の数）	-
0xBA03	8	標準レベル	SOAP クライアントライブラリーのサービスメソッド出口（ディスパッチベース）	クラス名	メソッド名（引数の数）	正常時 　- 異常時 　例外名
0xBA04	4	標準レベル	SOAP クライアントライブラリーの HTTP メッセージ送信前	クラス名	メソッド名（引数の数）	エンドポイント URI
0xBA05	7	標準レベル	SOAP クライアントライブラリーの HTTP メッセージ受信後	クラス名	メソッド名（引数の数）	正常時 　- 異常時 　例外名
0xBA06	5	標準レベル	SOAP サービス呼び出し前（Web サービス実装クラス）	クラス名	メソッド名（引数の数）	サービスメソッド名
0xBA07	6	標準レベル	SOAP サービス呼び出し後（Web サービス実装クラス）	クラス名	メソッド名（引数の数）	正常時 　- 異常時 　例外名※
0xBA08	5	標準レベル	SOAP サービス呼び出し前（プロバイダー実装クラス）	クラス名	メソッド名（引数の数）	-

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBA09	6	標準レベル	SOAP サービス呼び出し後（プロバイダー実装クラス）	クラス名	メソッド名（引数の数）	正常時 - 異常時 例外名※

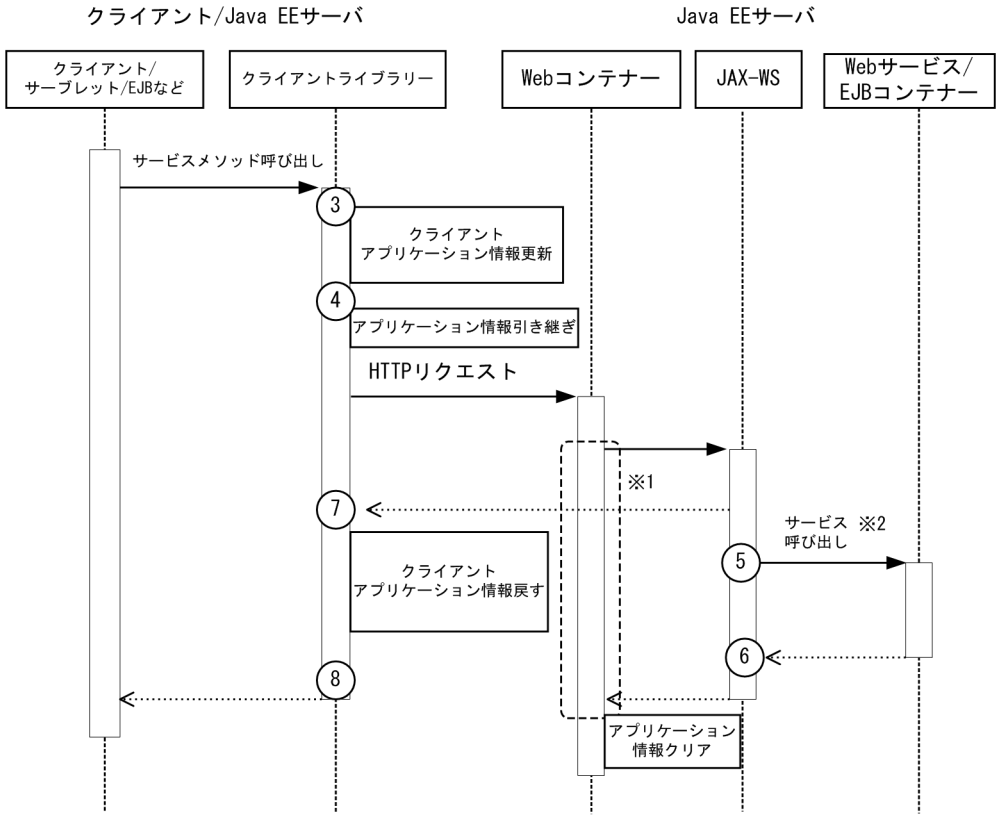
注※

トレース取得ポイントで取得した例外が java.lang.reflect.InvocationTargetException だった場合、java.lang.reflect.InvocationTargetException が保持している例外の例外名が出力されます。

one -way オペレーションの場合

one -way オペレーションの場合の、JAX-WS でのトレースの取得ポイントを次に示します。

図 9-26　JAX-WS のトレース取得ポイント（one -way オペレーションの場合）



（凡例）

(n)：トレース取得ポイントを示します。

メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

注※1

点線で囲まれた部分は、同期処理の場合の Web コンテナーのトレース取得ポイントと同じです。

注※2

EJB コンテナーは、EJB を Web サービスとして実装している場合に呼び出されます。

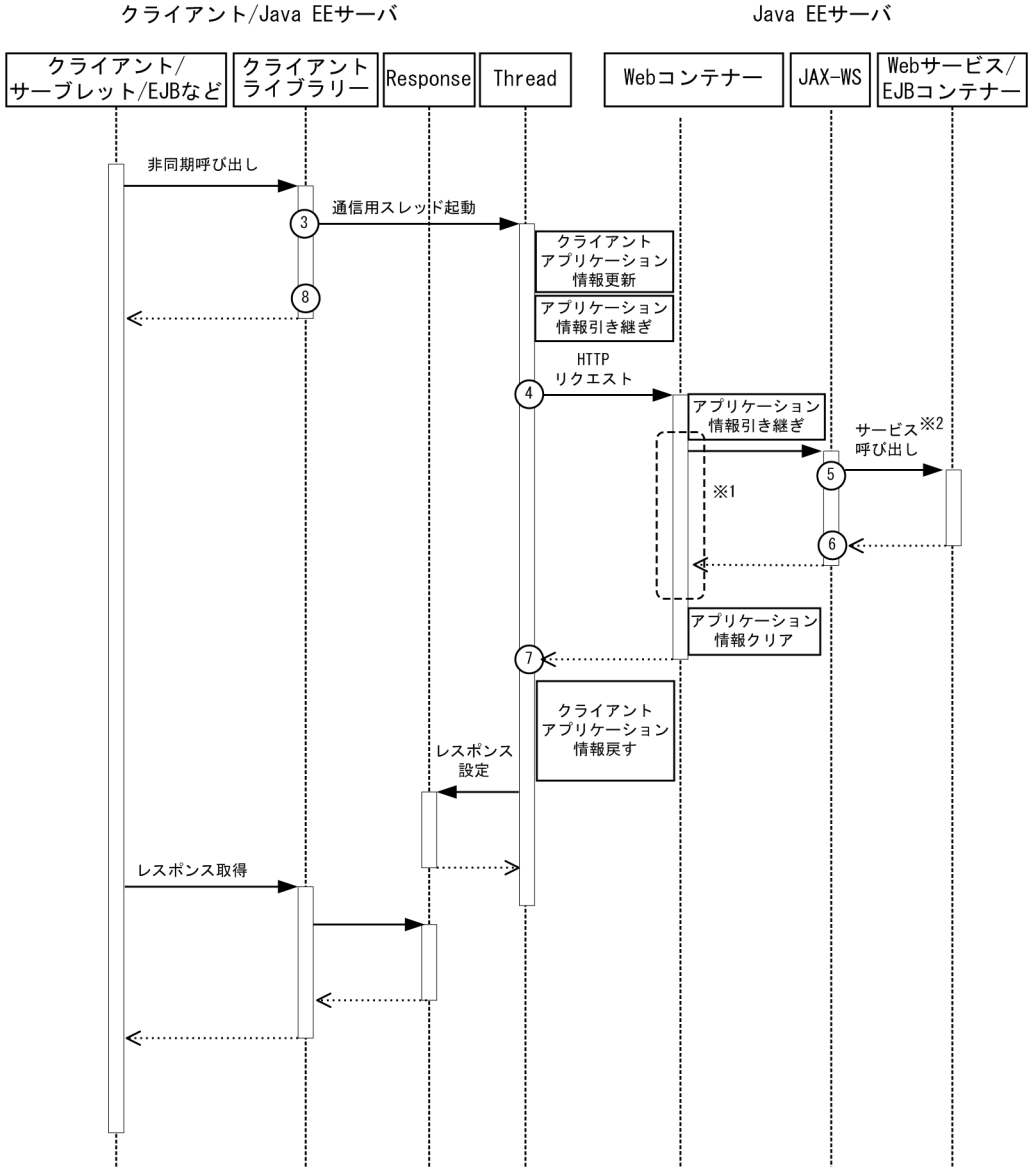
EJB コンテナー以降の取得個所は、Session Bean および Entity Bean のローカル呼び出しのときと同じです。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報については、JAX-WS のトレース取得ポイントの詳細（request-response オペレーション（同期）の場合）と同じです。

request-response オペレーション（非同期）の場合

request-response オペレーション（非同期）の場合の、 JAX-WS でのトレースの取得ポイントを次に示します。

図 9-27　JAX-WS のトレース取得ポイント（request-response オペレーション（非同期）の場合）



（凡例）

(n)：トレース取得ポイントを示します。

メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

注※1

点線で囲まれた部分は、Web コンテナーのトレース取得ポイント（同期処理の場合）と同じです。

注※2

EJB コンテナーは、EJB を Web サービスとして実装している場合に呼び出されます。

EJB コンテナー以降の取得個所は、Session Bean および Entity Bean（ローカル呼び出し）の場合と同じです。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報については、JAX-WS のトレース取得ポイントの詳細（request-response オペレーション（同期）の場合）と同じです。

関連項目

- 9.3.2　Web コンテナーのトレース取得ポイント
- 9.3.3　EJB コンテナーのトレース取得ポイント

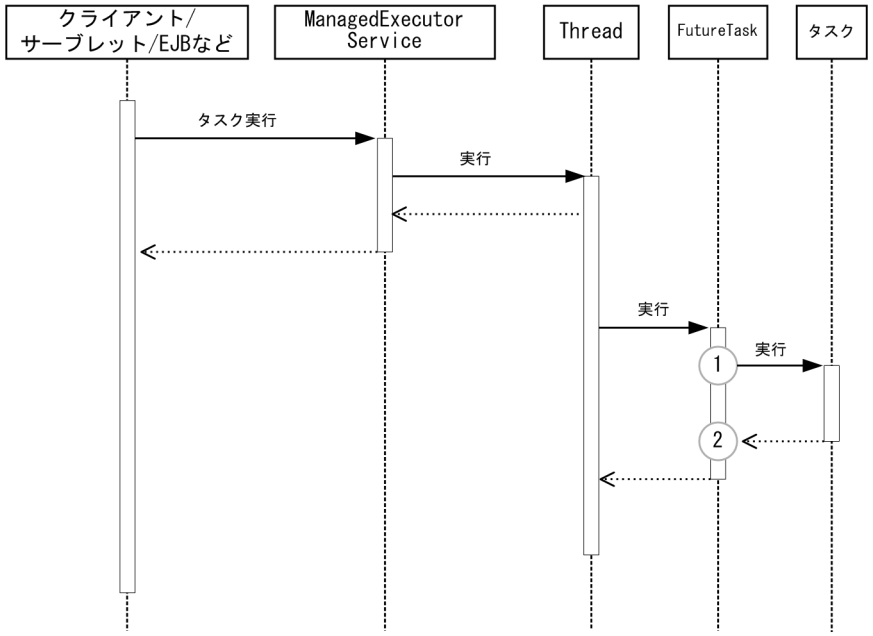
9.3.11　Concurrency Utilities のトレース取得ポイント

Concurrency Utilities でのトレース取得ポイントの詳細について説明します。

Java EE の ExecutorService 上でタスクを実行した場合

Java EE の ExecutorService 上でタスクを実行した場合の、Concurrency Utilities でのトレースの取得ポイントを次に示します。

図 9-28　Concurrency Utilities のトレース取得ポイント（Java EE の ExecutorService 上でタスクを実行した場合）



（凡例）

(n)：トレース取得ポイントを示します。

メモ

この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

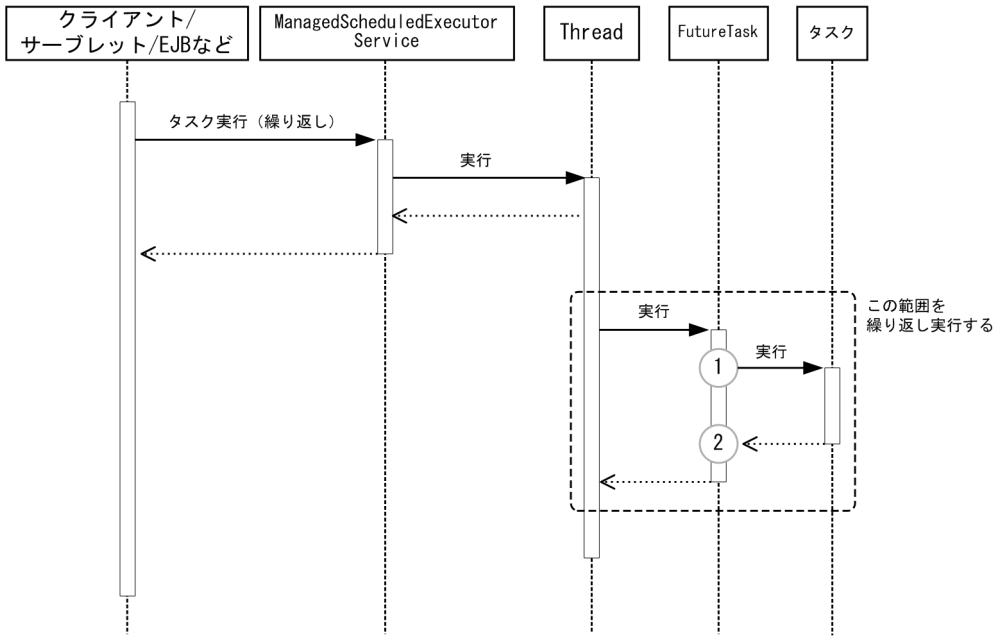
表 9-27　Concurrency Utilities のトレース取得ポイントの詳細（Java EE の ExecutorService 上でタスクを実行した場合）

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBC00	1	標準レベル	ManagedExecutorService または ManagedScheduledExecutorService でタスクを実行する直前	クラス名	-	-
0xBC01	2	標準レベル	ManagedExecutorService または ManagedScheduledExecutorService でタスクを実行した直後	クラス名	-	正常時 - 異常時 例外名

Java EE の ScheduledExecutorService 上でタスクを繰り返し実行した場合

Java EE の ScheduledExecutorService 上でタスクを繰り返し実行した場合 の、Concurrency Utilities でのトレースの取得ポイントを次に示します。

図 9-29　Concurrency Utilities のトレース取得ポイント（Java EE の ScheduledExecutorService 上でタスクを繰り返し実行した場合）



（凡例）

(n)：トレース取得ポイントを示します。

メモ

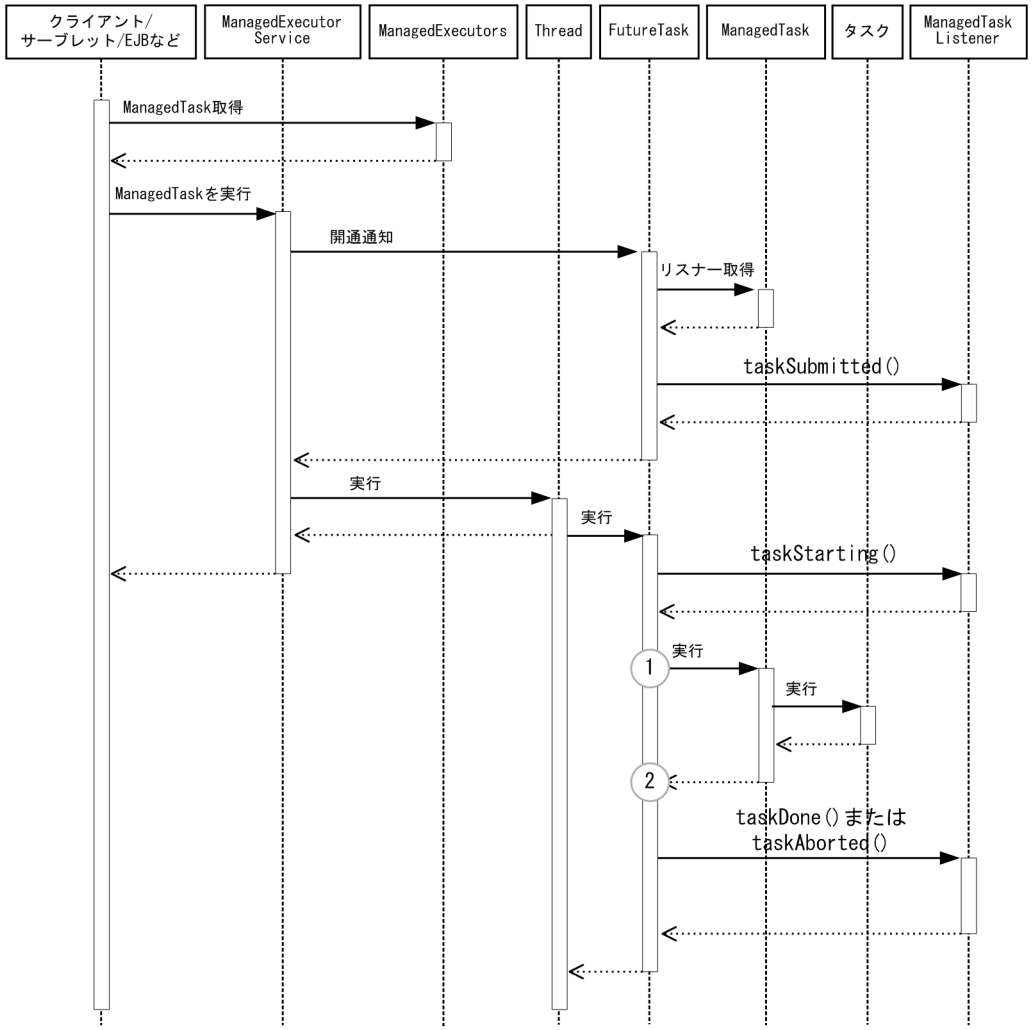
この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報については、Concurrency Utilities のトレース取得ポイントの詳細（Java EE の ExecutorService 上でタスクを実行した場合）と同じです。

Java EE の ExecutorService 上で ManagedTask を実行した場合

Java EE の ExecutorService 上で ManagedTask を実行した場合の、Concurrency Utilities でのトレースの取得ポイントを次に示します。

図 9-30　Concurrency Utilities のトレース取得ポイント（Java EE の ExecutorService 上で ManagedTask を実行した場合）



（凡例）

(n)：トレース取得ポイントを示します。

メモ

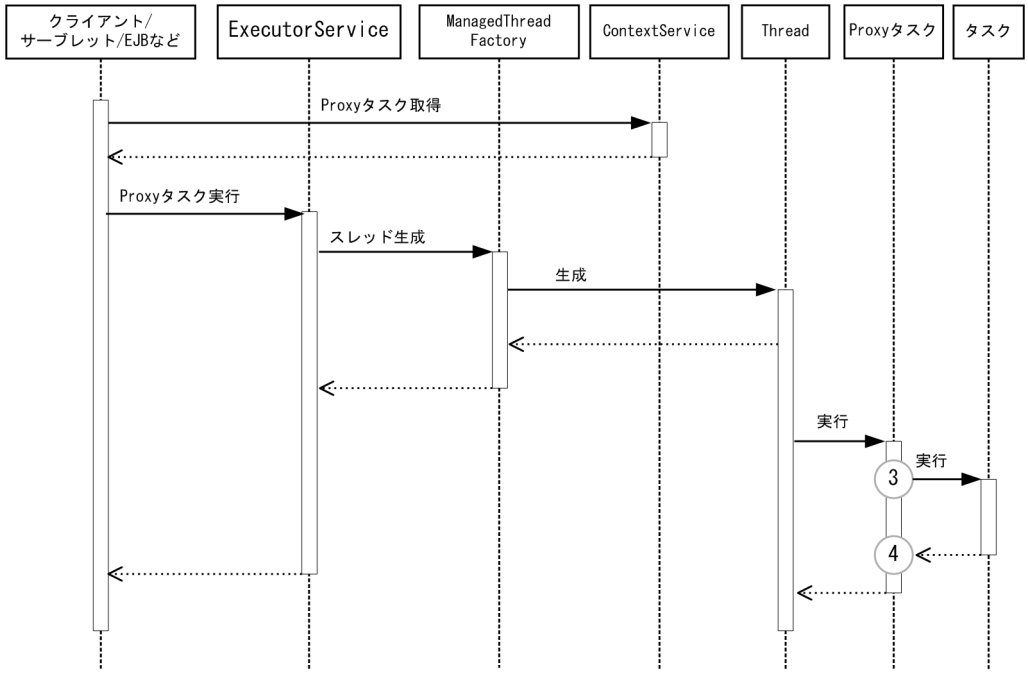
この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報については、Concurrency Utilities のトレース取得ポイントの詳細（Java EE の ExecutorService 上でタスクを実行した場合）と同じです。

Java SE の ExecutorService 上でコンテキスト情報を付加したタスクを実行した場合

Java SE の ExecutorService 上でコンテキスト情報を付加したタスクを実行した場合 の、Concurrency Utilities でのトレースの取得ポイントを次に示します。

図 9-31　Concurrency Utilities のトレース取得ポイント（Java SE の ExecutorService 上でコンテキスト情報を付加したタスクを実行した場合）



（凡例）

ⓝ：トレース取得ポイントを示します。

> **メモ**
>
> この図は、リクエストの延長で取得する流れを示す図です。トレースを取得する契機すべてを示す図ではありません。このほかにも取得する契機はあります。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-28　Concurrency Utilities のトレース取得ポイントの詳細（Java SE の ExecutorService 上でコンテキスト情報を付加したタスクを実行した場合）

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBC02	3	標準レベル	コンテキスト情報を付加したタスクを実行する直前	クラス名	-	-
0xBC03	4	標準レベル	コンテキスト情報を付加したタスクを実行した直後	クラス名	-	正常時 - 異常時 例外名

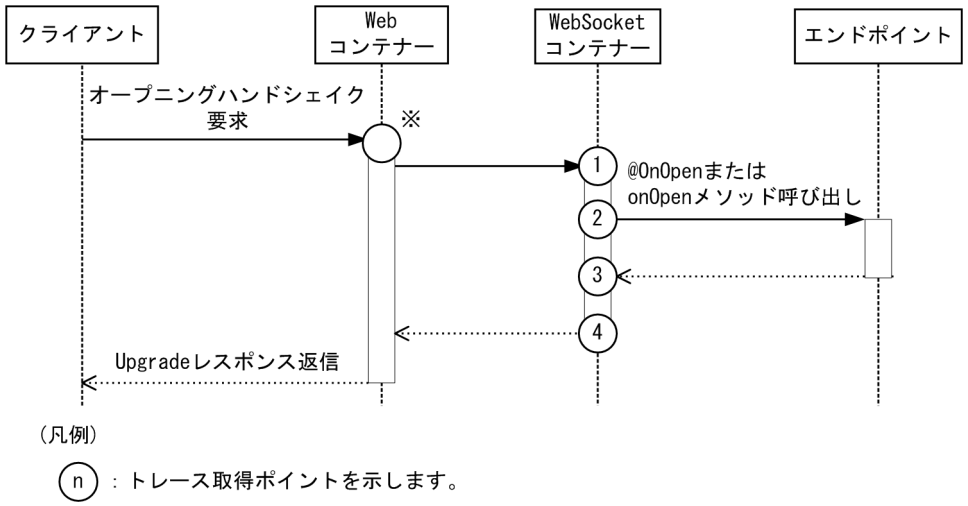
9.3.12　WebSocket のトレース取得ポイント

WebSocket でのトレース取得ポイントの詳細について説明します。

オープニングハンドシェイク要求受信時

WebSocket でのトレースの取得ポイントを次に示します。

図 9-32　WebSocket のトレース取得ポイント（オープニングハンドシェイク要求受信時）



注※

Web コンテナーの取得ポイント（0xB100）を示します。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-29　WebSocket のトレース取得ポイントの詳細（オープニングハンドシェイク要求受信時）

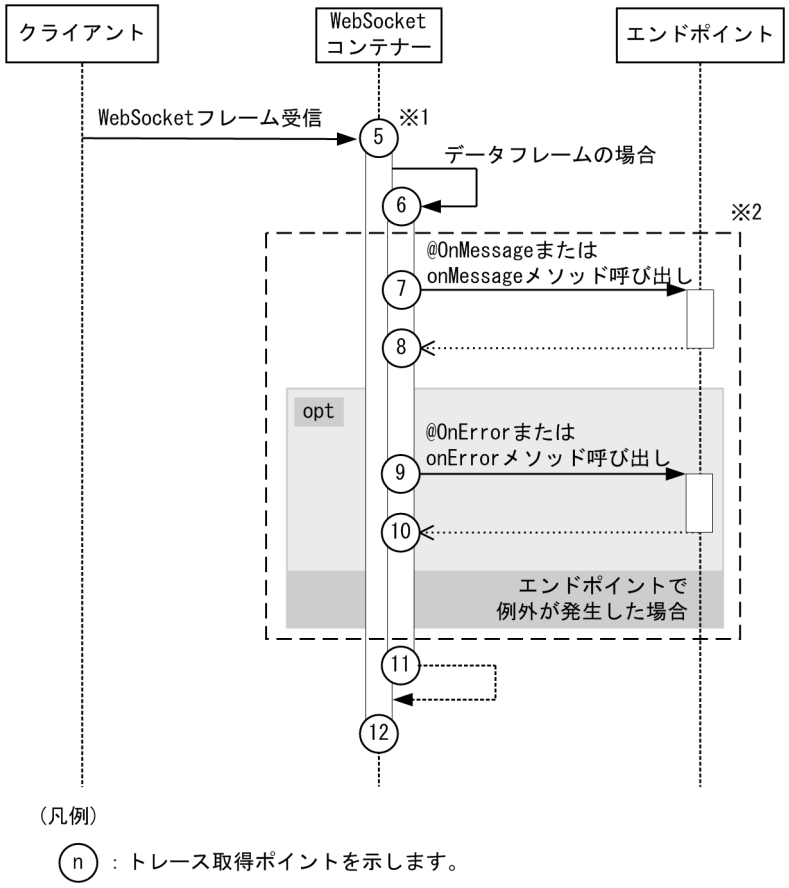
イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBD00	1	標準レベル	オープニングハンドシェイク要求受信後処理開始直前	CONNECT	ハンドシェイク時の URI	-
0xBD01	4	標準レベル	オープニングハンドシェイク要求受信後処理完了直後	CONNECT	ハンドシェイク時の URI	正常時 セッション ID 異常時 例外名
0xBD20	2	標準レベル	@OnOpen アノテーションが付与されたメソッドまたは javax.websocket.Endpoint 実装クラスの	エンドポイントのクラス名	メソッド名	-

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
			onOpen メソッドの呼び出し直前			
0xBD21	3	標準レベル	@OnOpen アノテーションが付与されたメソッドまたは javax.websocket.Endpoint 実装クラスの onOpen メソッドの処理完了直後	エンドポイントのクラス名	メソッド名	正常時 - 異常時 例外名

メッセージ受信時

WebSocket でのトレースの取得ポイントを次に示します。

図 9-33　WebSocket のトレース取得ポイント（メッセージ（データフレーム）受信時）



注※1

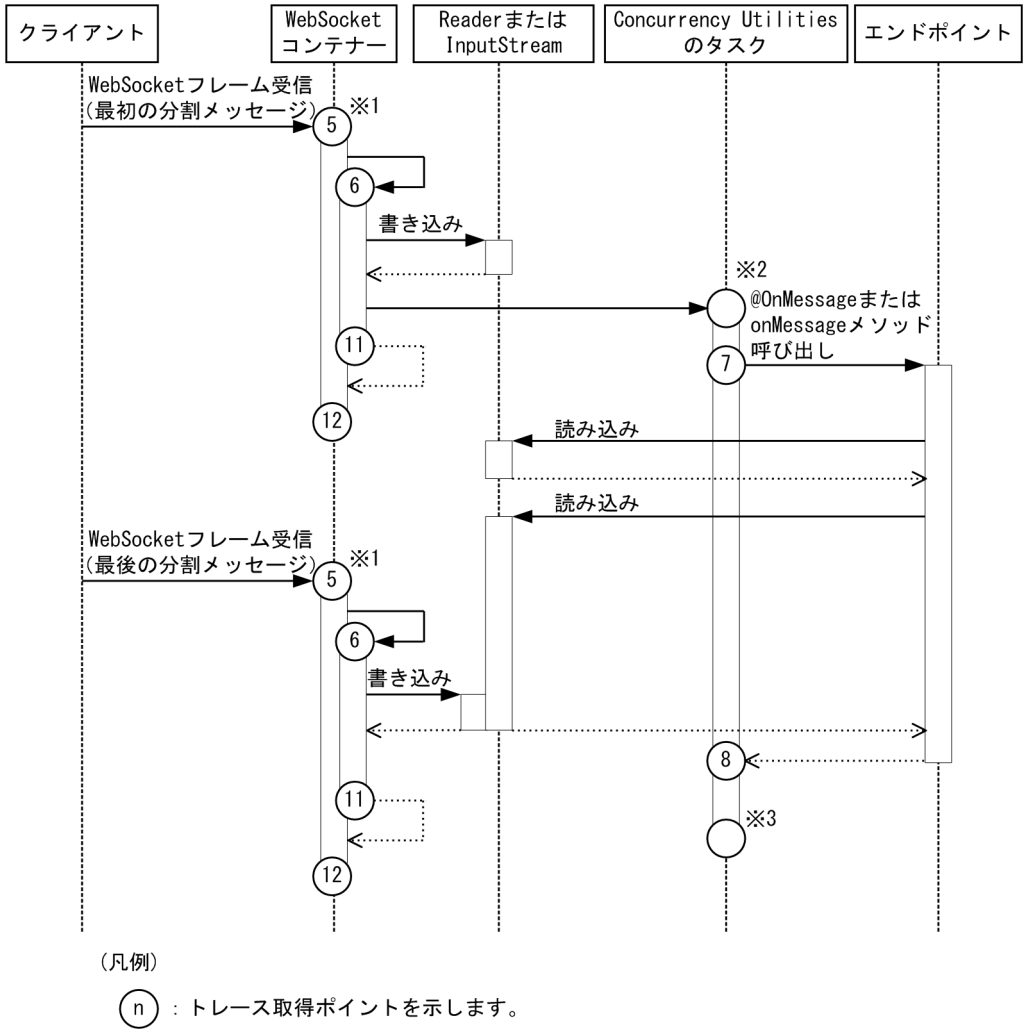
クライアントアプリケーション情報は新規採番されます。

ルートアプリケーション情報は、オープニングハンドシェイク時にクライアントに採番されたルートアプリケーション情報が適用されます。

注※2

エンドポイントで@OnMessage が付与されたメソッドまたはjavax.websocket.MessageHandler.Whole の実装クラスが使用されていて、分割されたメッセージを受信した場合、枠内の取得ポイントは最後のフレームを受信したときだけ呼ばれます。

図 9-34　WebSocket のトレース取得ポイント（Reader または InputStream 型の引数を持つメッセージハンドラによる分割メッセージ受信時）



注※1

クライアントアプリケーション情報は新規採番されます。

ルートアプリケーション情報は、オープニングハンドシェイク時にクライアントに採番されたルートアプリケーション情報が適用されます。

注※2

Concurrency Utilities の取得ポイント（0xBC00）を示します。

注※3

Concurrency Utilities の取得ポイント（0xBC01）を示します。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-30　WebSocket のトレース取得ポイントの詳細（メッセージ受信時）

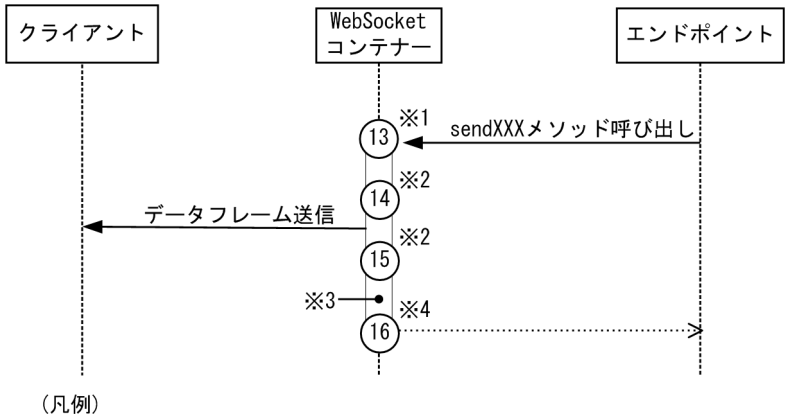
イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBD02	5	標準レベル	受信した WebSocket フレームに対する処理開始直前	READ	-	-
0xBD03	12	標準レベル	受信した WebSocket フレームに対する処理完了直後	READ	-	正常時 - 異常時 例外名
0xBD40	6	標準レベル	データフレーム受信後処理開始直前	MESSAGE	次のどれか • onMessage(String) • onMessage(ByteBuffer) • onPartialMessage(String) • onPartialMessage(ByteBuffer)	-
0xBD41	11	標準レベル	データフレーム受信後処理完了直後	MESSAGE	次のどれか • onMessage(String) • onMessage(ByteBuffer) • onPartialMessage(String) • onPartialMessage(ByteBuffer)	正常時 - 異常時 例外名
0xBD22	7	標準レベル	@OnMessage アノテーションが付与されたメソッドの呼び出し直前	エンドポイントのクラス名	メソッド名	-

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBD23	8	標準レベル	@OnMessage アノテーションが付与されたメソッド処理完了直後	エンドポイントのクラス名	メソッド名	正常時 - 異常時 例外名
0xBD24	7	標準レベル	javax.websocket.MessageHandler.Whole の実装クラスの onMessage メソッド呼び出し直前	javax.websocket.MessageHandler.Whole の実装クラス名	メソッド名	-
0xBD25	8	標準レベル	javax.websocket.MessageHandler.Whole の実装クラスの onMessage メソッド処理完了直後	javax.websocket.MessageHandler.Whole の実装クラス名	メソッド名	正常時 - 異常時 例外名
0xBD26	7	標準レベル	javax.websocket.MessageHandler.Partial の実装クラスの onMessage メソッド呼び出し直前	javax.websocket.MessageHandler.Partial の実装クラス名	メソッド名	-
0xBD27	8	標準レベル	javax.websocket.MessageHandler.Partial の実装クラスの onMessage メソッド処理完了直後	javax.websocket.MessageHandler.Partial の実装クラス名	メソッド名	正常時 - 異常時 例外名
0xBD28	9	標準レベル	@OnError アノテーションが付与されたメソッドまたは javax.websocket.Endpoint 実装クラスの onError メソッドの呼び出し直前	エンドポイントのクラス名	メソッド名	-
0xBD29	10	標準レベル	@OnError アノテーションが付与されたメソッドまたは javax.websocket.Endpoint 実装クラスの onError メソッドの処理完了直後	エンドポイントのクラス名	メソッド名	正常時 - 異常時 例外名

データ送信時

WebSocket でのトレースの取得ポイントを次に示します。

図 9-35　WebSocket のトレース取得ポイント（データ送信時）



注※1

クライアントアプリケーション情報は新規採番されます。

ルートアプリケーション情報は、sendXXX メソッドの呼び出し元のルートアプリケーション情報が適用されます。

注※2

クライアントアプリケーション情報は、項番 13 で採番されたクライアントアプリケーション情報が適用されます。

ルートアプリケーション情報は、データフレーム送信先のルートアプリケーション情報が適用されます。

注※3

ここで WebSocket アクセスログが出力されます。

ルートアプリケーション情報は項番 15 と同じ値が適用されます。

注※4

クライアントアプリケーション情報は、項番 13 で採番されたクライアントアプリケーション情報が適用されます。

ルートアプリケーション情報は、sendXXX メソッドの呼び出し元のルートアプリケーション情報が適用されます。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-31　WebSocket のトレース取得ポイントの詳細（データ送信時）

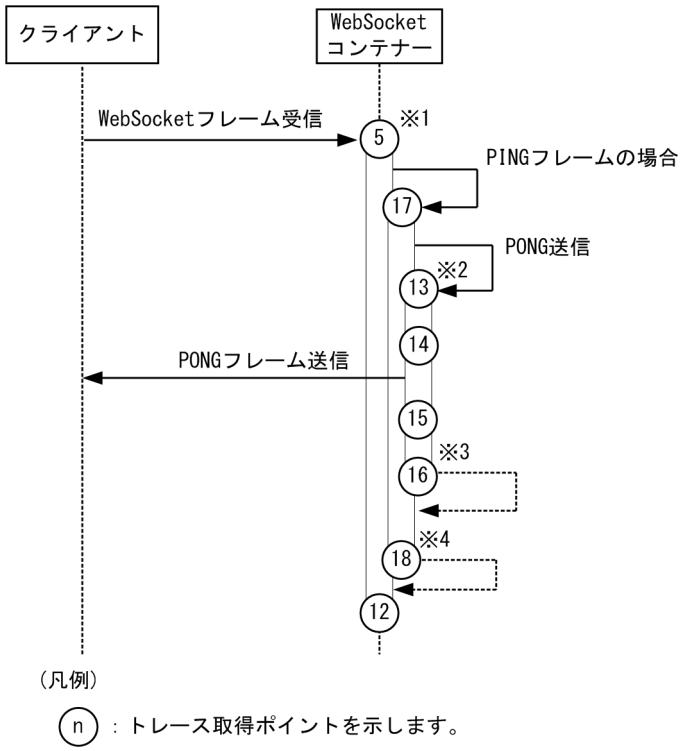
イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBD12	14	標準レベル	WebSocket フレームの送信処理開始直前	WRITE	-	-

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBD13	15	標準レベル	WebSocket フレームの送信処理完了直後	WRITE	-	正常時 - 異常時 例外名
0xBD30	13	標準レベル	javax.websocket.RemoteEndpoint.Async.sendXXX メソッドが呼び出された直後	javax.websocket.RemoteEndpoint.Async	メソッド名(引数の型)	送信先のセッション ID
0xBD31	16	標準レベル	javax.websocket.RemoteEndpoint.Async.sendXXX メソッドのリターン直前	javax.websocket.RemoteEndpoint.Async	メソッド名(引数の型)	正常時 - 異常時 例外名
0xBD32	13	標準レベル	javax.websocket.RemoteEndpoint.Basic.sendXXX メソッドが呼び出された直後	javax.websocket.RemoteEndpoint.Basic	メソッド名(引数の型)	送信先のセッション ID
0xBD33	16	標準レベル	javax.websocket.RemoteEndpoint.Basic.sendXXX メソッドのリターン直前	javax.websocket.RemoteEndpoint.Basic	メソッド名(引数の型)	正常時 - 異常時 例外名
0xBD34	13	標準レベル	javax.websocket.RemoteEndpoint.sendXXX メソッドが呼び出された直後	javax.websocket.RemoteEndpoint	メソッド名(引数の型)	送信先のセッション ID
0xBD35	16	標準レベル	javax.websocket.RemoteEndpoin.sendXXX メソッドのリターン直前	javax.websocket.RemoteEndpoint	メソッド名(引数の型)	正常時 - 異常時 例外名

PING 受信時

WebSocket でのトレースの取得ポイントを次に示します。

図 9-36　WebSocket のトレース取得ポイント（PING 受信時）



注※1

クライアントアプリケーション情報は新規採番されます。

ルートアプリケーション情報は、オープニングハンドシェイク時にクライアントに採番されたルートアプリケーション情報が適用されます。

注※2

クライアントアプリケーション情報は新規採番されます。

ルートアプリケーション情報は、項番 5 と同じルートアプリケーション情報が適用されます。

注※3

クライアントアプリケーション情報は、項番 13 で採番されたクライアントアプリケーション情報が適用されます。

ルートアプリケーション情報は、項番 5 と同じルートアプリケーション情報が適用されます。

注※4

クライアントアプリケーション情報は、項番 5 で採番したクライアントアプリケーション情報が適用されます。

ルートアプリケーション情報は、項番 5 と同じルートアプリケーション情報が適用されます。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-32　WebSocket のトレース取得ポイントの詳細（PING 受信時）

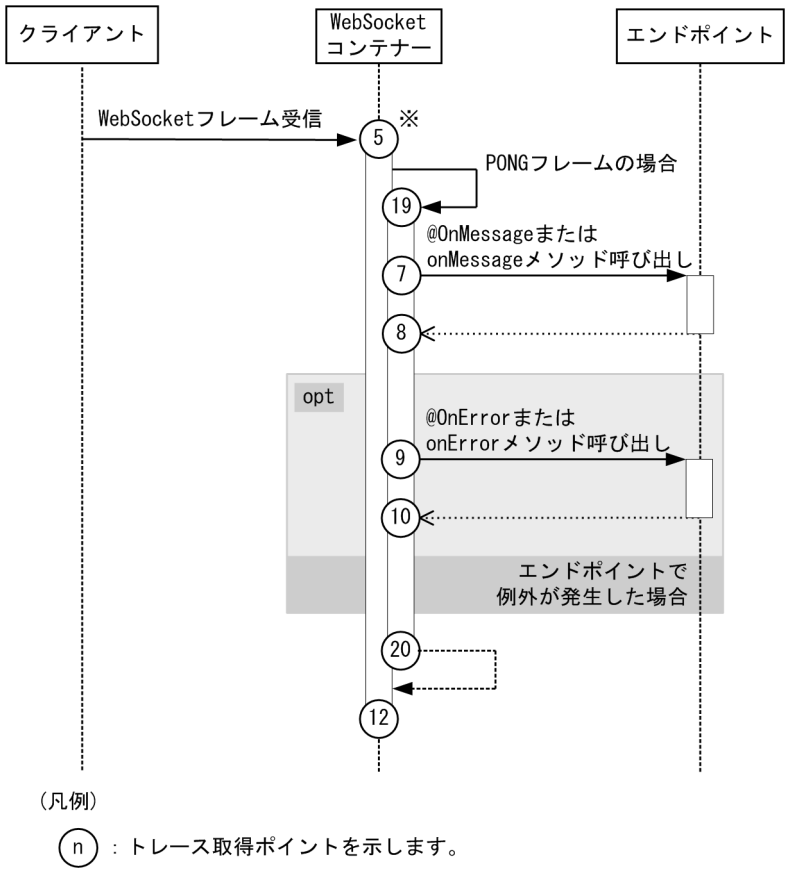
イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBD02	5	標準レベル	受信した WebSocket フレームに対する処理開始直前	READ	-	-
0xBD03	12	標準レベル	受信した WebSocket フレームに対する処理完了直後	READ	-	正常時 　- 異常時 　例外名
0xBD12	14	標準レベル	WebSocket フレームの送信処理開始直前	WRITE	-	-
0xBD13	15	標準レベル	WebSocket フレームの送信処理完了直後	WRITE	-	正常時 　- 異常時 　例外名
0xBD42	17	標準レベル	PING フレーム受信後処理開始直前	PING	-	-
0xBD43	18	標準レベル	PING フレーム受信後処理完了直後	PING	-	正常時 　- 異常時 　例外名
0xBD30	13	標準レベル	javax.websocket.RemoteEndpoint.Async.sendXXX メソッドが呼び出された直後	javax.websocket.RemoteEndpoint.Async	メソッド名(引数の型)	送信先のセッション ID
0xBD31	16	標準レベル	javax.websocket.RemoteEndpoint.Async.sendXXX メソッドのリターン直前	javax.websocket.RemoteEndpoint.Async	メソッド名(引数の型)	正常時 　- 異常時 　例外名
0xBD32	13	標準レベル	javax.websocket.RemoteEndpoint.Basic.sendXXX メソッドが呼び出された直後	javax.websocket.RemoteEndpoint.Basic	メソッド名(引数の型)	送信先のセッション ID
0xBD33	16	標準レベル	javax.websocket.RemoteEndpoint.Basic.sendXXX メソッドのリターン直前	javax.websocket.RemoteEndpoint.Basic	メソッド名(引数の型)	正常時 　-

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
						異常時 例外名
0xBD34	13	標準レベル	javax.websocket.RemoteEndpoint.sendXXX メソッドが呼び出された直後	javax.websocket.RemoteEndpoint	メソッド名 (引数の型)	送信先のセッション ID
0xBD35	16	標準レベル	javax.websocket.RemoteEndpoin.sendXXX メソッドのリターン直前	javax.websocket.RemoteEndpoint	メソッド名 (引数の型)	正常時 - 異常時 例外名

PONG 受信時

WebSocket でのトレースの取得ポイントを次に示します。

図 9-37　WebSocket のトレース取得ポイント（PONG 受信時）



注※

クライアントアプリケーション情報は新規採番されます。

ルートアプリケーション情報は、オープニングハンドシェイク時にクライアントに採番されたルートアプリケーション情報が適用されます。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-33　WebSocket のトレース取得ポイントの詳細（PONG 受信時）

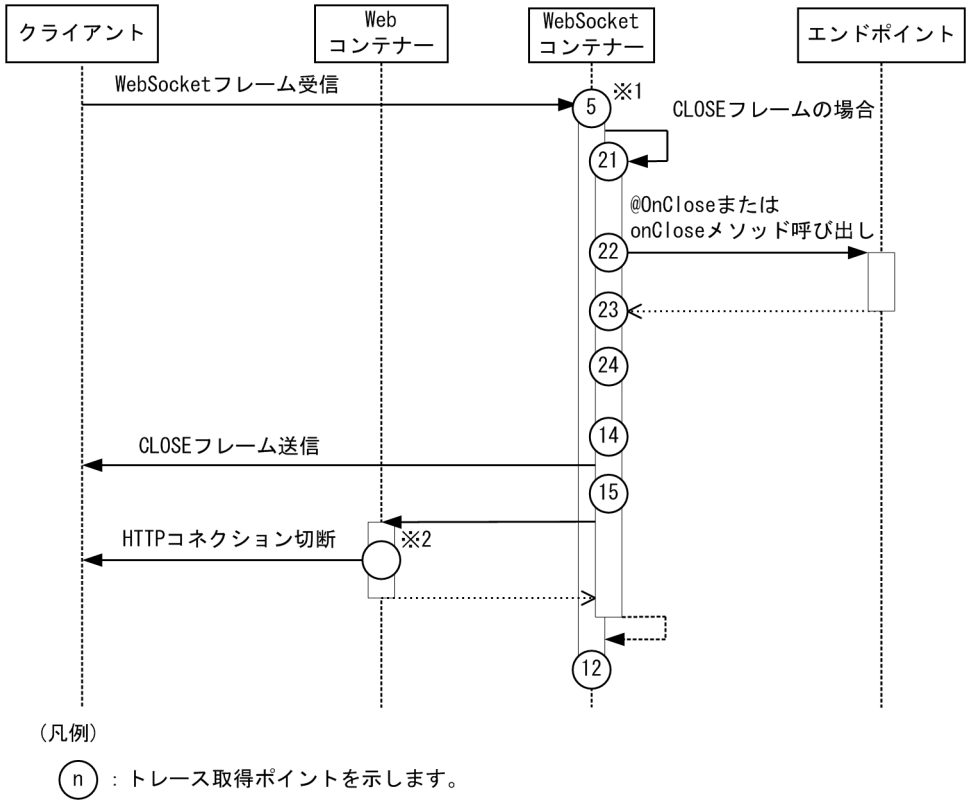
イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBD02	5	標準レベル	受信した WebSocket フレームに対する処理開始直前	READ	-	-
0xBD03	12	標準レベル	受信した WebSocket フレームに対する処理完了直後	READ	-	正常時 - 異常時 例外名
0xBD44	19	標準レベル	PONG フレーム受信後処理開始直前	PONG	-	-
0xBD45	20	標準レベル	PONG フレーム受信後処理完了直後	PONG	-	正常時 - 異常時 例外名
0xBD22	7	標準レベル	@OnMessage アノテーションが付与されたメソッドの呼び出し直前	エンドポイントのクラス名	メソッド名	-
0xBD23	8	標準レベル	@OnMessage アノテーションが付与されたメソッド処理完了直後	エンドポイントのクラス名	メソッド名	正常時 - 異常時 例外名
0xBD24	7	標準レベル	javax.websocket.MessageHandler.Whole の実装クラスの onMessage メソッド呼び出し直前	javax.websocket.MessageHandler.Whole の実装クラス名	メソッド名	-
0xBD25	8	標準レベル	javax.websocket.MessageHandler.Whole の実装クラスの onMessage メソッド処理完了直後	javax.websocket.MessageHandler.Whole の実装クラス名	メソッド名	正常時 - 異常時 例外名

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBD26	7	標準レベル	javax.websocket.MessageHandler.Partial の実装クラスの onMessage メソッド呼び出し直前	javax.websocket.MessageHandler.Partial の実装クラス名	メソッド名	-
0xBD27	8	標準レベル	javax.websocket.MessageHandler.Partial の実装クラスの onMessage メソッド処理完了直後	javax.websocket.MessageHandler.Partial の実装クラス名	メソッド名	正常時 - 異常時 例外名
0xBD28	9	標準レベル	@OnError アノテーションが付与されたメソッドまたは javax.websocket.Endpoint 実装クラスの onError メソッドの呼び出し直前	エンドポイントのクラス名	メソッド名	-
0xBD29	10	標準レベル	@OnError アノテーションが付与されたメソッドまたは javax.websocket.Endpoint 実装クラスの onError メソッドの処理完了直後	エンドポイントのクラス名	メソッド名	正常時 - 異常時 例外名

クロージングハンドシェイク要求受信時

WebSocket でのトレースの取得ポイントを次に示します。

図 9-38　WebSocket のトレース取得ポイント（クロージングハンドシェイク要求受信時（ペイロードデータあり））



注※1

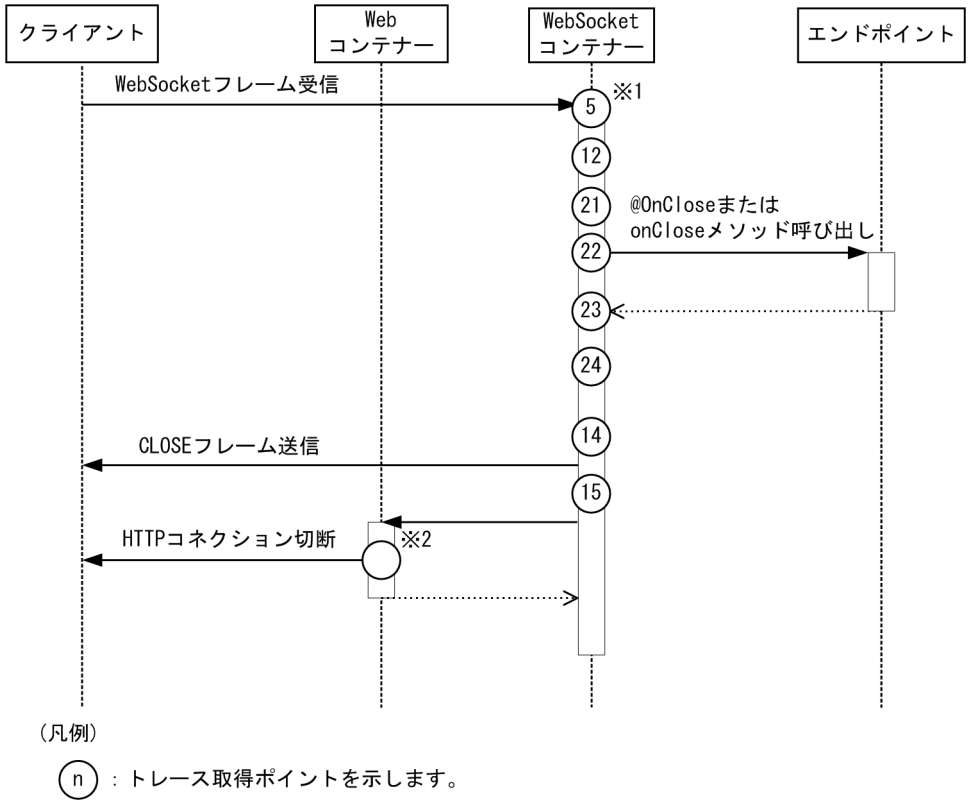
クライアントアプリケーション情報は新規採番されます。

ルートアプリケーション情報は、オープニングハンドシェイク時にクライアントに採番されたルートアプリケーション情報が適用されます。

注※2

Web コンテナーの取得ポイント（0xB101）を示します。

図 9-39　WebSocket のトレース取得ポイント（クロージングハンドシェイク要求受信時（ペイロードデータなし））



注※1

クライアントアプリケーション情報は新規採番されます。

ルートアプリケーション情報は、オープニングハンドシェイク時にクライアントに採番されたルートアプリケーション情報が適用されます。

注※2

Web コンテナーの取得ポイント（0xB101）を示します。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-34　WebSocket のトレース取得ポイントの詳細（クロージングハンドシェイク要求受信時）

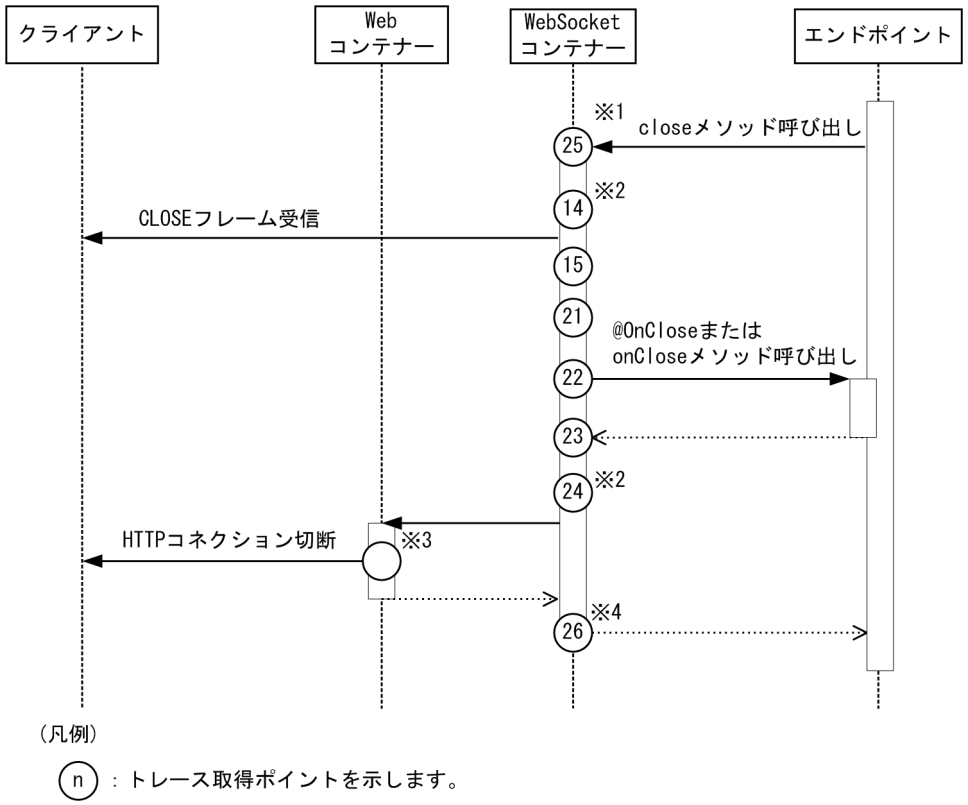
イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBD02	5	標準レベル	受信した WebSocket フレームに対する処理開始直前	READ	-	-
0xBD03	12	標準レベル	受信した WebSocket フレー	READ	-	正常時 -

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
			ムに対する処理完了直後			異常時 例外名
0xBD12	14	標準レベル	WebSocket フレームの送信処理開始直前	WRITE	-	-
0xBD13	15	標準レベル	WebSocket フレームの送信処理完了直後	WRITE	-	正常時 - 異常時 例外名
0xBD46	21	標準レベル	クロージングハンドシェイク要求受信後処理または送信後処理開始直前	CLOSE	切断要因	-
0xBD47	24	標準レベル	クロージングハンドシェイク要求受信後処理または送信後処理完了直後	CLOSE	切断要因	正常時 - 異常時 例外名
0xBD2A	22	標準レベル	@OnClose アノテーションが付与されたメソッドまたは javax.websocket.Endpoint 実装クラスの onClose メソッドの呼び出し直前	エンドポイントのクラス名	メソッド名	-
0xBD2B	23	標準レベル	@OnClose アノテーションが付与されたメソッドまたは javax.websocket.Endpoint 実装クラスの onClose メソッドの処理完了直後	エンドポイントのクラス名	メソッド名	正常時 - 異常時 例外名

クロージングハンドシェイク要求送信時

WebSocket でのトレースの取得ポイントを次に示します。

図 9-40　WebSocket のトレース取得ポイント（クロージングハンドシェイク要求送信時）



注※1

クライアントアプリケーション情報は新規採番されます。

ルートアプリケーション情報は、sendXXX メソッドの呼び出し元のルートアプリケーション情報が適用されます。

注※2

クライアントアプリケーション情報は、項番 25 で採番されたクライアントアプリケーション情報が適用されます。

ルートアプリケーション情報は、データフレーム送信先のルートアプリケーション情報が適用されます。

注※3

Web コンテナーの取得ポイント（0xB101）を示します。

注※4

クライアントアプリケーション情報は、項番 25 で採番されたクライアントアプリケーション情報が適用されます。

ルートアプリケーション情報は、sendXXX メソッドの呼び出し元のルートアプリケーション情報が適用されます。

イベント ID、トレースレベル、トレース取得ポイント、取得できる情報について次の表に示します。

表 9-35　WebSocket のトレース取得ポイントの詳細（クロージングハンドシェイク要求送信時

イベント ID	図中の番号	PRF トレース取得レベル	トレース取得ポイント	取得できる情報		
				インターフェース名	オペレーション名	オプション
0xBD12	14	標準レベル	WebSocket フレームの送信処理開始直前	WRITE	-	-
0xBD13	15	標準レベル	WebSocket フレームの送信処理完了直後	WRITE	-	正常時 - 異常時 例外名
0xBD46	21	標準レベル	クロージングハンドシェイク要求受信後処理または送信後処理開始直前	CLOSE	切断要因	-
0xBD47	24	標準レベル	クロージングハンドシェイク要求受信後処理または送信後処理完了直後	CLOSE	切断要因	正常時 - 異常時 例外名
0xBD2A	22	標準レベル	@OnClose アノテーションが付与されたメソッドまたは javax.websocket.Endpoint 実装クラスの onClose メソッドの呼び出し直前	エンドポイントのクラス名	メソッド名	-
0xBD2B	23	標準レベル	@OnClose アノテーションが付与されたメソッドまたは javax.websocket.Endpoint 実装クラスの onClose メソッドの処理完了直後	エンドポイントのクラス名	メソッド名	正常時 - 異常時 例外名
0xBD36	25	標準レベル	javax.websocket.Session.close メソッドが呼び出された直後	javax.websocket.Session	メソッド名(引数の型)	送信先のセッション ID
0xBD37	26	標準レベル	javax.websocket.Session.close メソッドのリターン直前	javax.websocket.Session	メソッド名(引数の型)	正常時 - 異常時 例外名

用語解説

英字

Administration Console

Application Server を使用したシステムの構築・運用を行うための GUI です。

CA

Certificate Authority の略称です。認証局と呼ばれる、SSL を使用するための証明書を発行する機関です。ほかのエンティティのために有効な証明書を発行する、信頼のおけるエンティティです。次に示す種類があります。

ルート CA：上位の CA による認証を受けないで、自分の正当性を自ら証明します。

中間 CA：自分の正当性を証明するために、上位の CA による認証を受けています。

CopyGC

SerialGC 使用時に New 領域に対して行われる GC のことです。

Explicit ヒープ

明示管理ヒープ機能で使用する、GC の対象とならないメモリー領域のことです。

FullGC

Tenured 領域に対して行われる GC のことです。

G1GC

GC の処理方式の一つです。New 領域に YoungGC、Tenured 領域に MixedGC を行います。GC によるアプリケーションの停止時間を重視した処理方式です。

GC

JavaVM が自動で行うメモリの回収処理のことです。

HTTP

HyperText Transfer Protocol の略称です。インターネットで、Web サーバと Web クライアントの間で HTML 文書を送受信するための通信プロトコルです。

Java EE RI

Java EE Reference Implementation の略称です。Java EE 仕様の参照実装です。

Java EE RI の DD

Java EE 仕様の参照実装が提供する Deployment Descriptor です。

SerialGC

GC の処理方式の一つです。New 領域に CopyGC、Tenured 領域に FullGC を行います。スループットを重視した処理方式です。

SSL

Secure Sockets Layer の略称です。情報を暗号化してネットワークで送受信するためのプロトコルの一つです。TCP 層の上位層です。クライアントと Web サーバの間で、証明書による認証、鍵交換、暗号化およびメッセージ認証をします。

TLS

Transport Layer Security の略称です。情報を暗号化してネットワークで送受信するためのプロトコルの一つです。SSL を基に改良が加えられたプロトコルです。

URI

Uniform Resource Identifier の略称です。URL のように Web 上のデータの位置を指定する方法と、Web 上の特定のデータを固有の名前で指定する方法を持ち合わせた規格です。URL と URN を組み合わせたもので、XML 文書中でインターネット資源を表したり、名前空間を識別したりするときに使用します。URI は一般的概念で、URL の上位集合です。RFC1630 で規定されています。

Web サーバ

Web ブラウザーからのリクエスト受信および Web ブラウザーへのデータ送信に関連する処理を実行するプログラムです。

Web フロントシステム

Application Server が主な対象としているシステムパターンです。セキュリティー装置、ロードバランサー、Application Server、データベースサーバ、ジョブ管理サーバ/メッセージ監視サーバで構成されるシステムのことです。

カ行

拡張 DD

Java EE アプリケーションや Java EE モジュールごとに、Java EE Server 固有機能の設定を行うための、XML 形式の定義ファイルです。

クラスター

同じアプリケーション、リソース、および設定情報を共有するサーバインスタンスのグループです。

計画停止

実行中のサーバプロセスまたはサーバスレッドを、実行終了後に停止する方法です。

公開鍵

公開鍵暗号で使用する、データの暗号化または電子署名の検証をするための鍵です。公開鍵は、通常ネットワーク上などで公開されています。データを暗号化して送信する場合、送信者は受信者の公開鍵を使用してデータを暗号化します。また、受信した電子署名を検証する場合は、受信者は送信者の公開鍵を使用して電子署名を検証します。

サ行

サーバインスタンス

Java EE アプリケーションを稼働させるサーバプロセスのことです。

サーバテンプレート

サーバインスタンス、Web サーバ、およびパフォーマンストレーサーの運用に必要な設定を記述しているもので、ドメイン管理サーバが環境構築時に参照するファイルです。

設定対象識別子

`asadmin set/get/list` サブコマンドを使用して、設定値を変更または取得できる設定対象に対する識別子のことです。

タ行

ドメイン

業務システムを構成するサーバインスタンス、Web サーバ、またはパフォーマンストレーサーといったコンポーネントを管理するグループのことです。

ドメイン管理サーバ

ドメインの運用管理を行うサーバのことです。

ナ行

ノード

サーバインスタンス、Web サーバ、およびパフォーマンストレーサーが起動されるホストのことです。

ハ行

秘密鍵

公開鍵暗号で使用する、データの復号化または電子署名を作成するための鍵（private key）です。非公開鍵ともいいます。 秘密鍵は、その所有者が安全に管理します。 暗号化されたデー

タを受信した場合、受信者は自分の秘密鍵を使用してデータを復号化します。また、送信するデータに電子署名を付加する場合は、送信者は自分の秘密鍵を使用し電子署名を作成します。

マ行

明示管理ヒープ機能

SerialGC 使用時に、FullGC の発生を抑止する機能です。

ラ行

リモートモード

ドメイン管理サーバと通信してコマンドを動作させるモードです。

ローカルモード

ドメイン管理サーバと通信しないでコマンドを動作させるモードです。

ロードバランサー

バックエンドサーバの負荷を均等にするために、リクエスト転送を調整する装置またはソフトウェアです。

索引

D

E

F

G

H

I

J

L

M

P

R

S

U

V

W

Y

あ

い

ま

め

り

ろ

株式会社 日立製作所
〒 100-8280　東京都千代田区丸の内一丁目 6 番 6 号