

# JEUS セッション管理ガイド

JEUS v8.0



Copyright © 2016 TmaxSoft Co., Ltd. All Rights Reserved.

## Copyright Notice

Copyright © 2016 TmaxSoft Co., Ltd. All Rights Reserved.

45, Jeongjail-ro, Bundang-gu, Seongnam-si, Gyeonggi-do, 13613, South Korea

## Restricted Rights Legend

All TmaxSoft Software (JEUS®) and documents are protected by copyright laws and international convention. TmaxSoft software and documents are made available under the terms of the TmaxSoft License Agreement and this document may only be distributed or copied in accordance with the terms of this agreement. No part of this document may be transmitted, copied, deployed, or reproduced in any form or by any means, electronic, mechanical, or optical, without the prior written consent of TmaxSoft Co., Ltd.

Nothing in this software document and agreement constitutes a transfer of intellectual property rights regardless of whether or not such rights are registered) or any rights to TmaxSoft trademarks, logos, or any other brand features. This document is for information purposes only. The company assumes no direct or indirect responsibilities for the contents of this document, and does not guarantee that the information contained in this document satisfies certain legal or commercial conditions. The information contained in this document is subject to change without prior notice due to product upgrades or updates. The company assumes no liability for any errors in this document.

このソフトウェア(JEUS®)マニュアルの内容とプログラムは、日本国の著作権法および国際条約によって保護されています。マニュアルの内容とプログラムは、TmaxSoft Co., Ltd.との使用許諾契約書の下でのみ使用することができ、マニュアルは使用許諾契約で許可されている範囲を除いては、配布または複製することができません。TmaxSoftの書面による事前の承諾を得ることなく、このマニュアルの全部または一部を電子的または機械的な方法を問わず、転送、複製、配布したり、または二次的著作物を作成する等の行為を一切禁じます。

このソフトウェアのマニュアルとプログラムの使用許諾契約は、いかなる場合においても、マニュアル及びプログラムと関連する知的財産権(登録の有無を問わず)を譲渡するものと解釈されず、TmaxSoftのブランド、ロゴ、商標等の使用権限を与えるものではありません。マニュアルは、情報を提供する目的でのみ提供しており、これに伴う契約上の直接的ないしは間接的な責任を負わず、マニュアルの内容は法律上もしくは商業的な特定の条件が満たされることを保証しません。マニュアルの内容は、製品のアップグレード及び修正により、その内容が予告なく変更されることがあり、内容上の誤りがないことを保証しません。

## Trademarks

JEUS® is registered trademark of TmaxSoft Co., Ltd.

JEUS®は、TmaxSoft Co., Ltd.の登録商標です。

Java and Solaris are registered trademarks of Oracle Corporation and its subsidiaries and affiliates.

Java、Solarisは、Oracle Corporation及びその子会社、関連会社の登録商標です。

Microsoft, Windows, and Windows NT are registered trademarks or trademarks of Microsoft Corporation.

Microsoft、Windows、Windows NTは、Microsoft Corporationの登録商標または商標です。

HP-UX is a registered trademark of Hewlett Packard Enterprise Company.

HP-UXは、Hewlett Packard Enterprise Companyの登録商標です。

---

AIX is a registered trademark of International Business Machines Corporation.

AIXは、International Business Machines Corporationの登録商標です。

UNIX is a registered trademark of X/Open Company, Ltd.

UNIXは、X/Open Company, Ltd.の登録商標です。

Linux is a registered trademark of Linus Torvalds.

Linuxは、Linus Torvaldsの登録商標です。

Other products and company names are trademarks or registered trademarks of their respective owners.

その他、記載されている会社名、製品名などは、各社の商号、商標または登録商標です。

The names of companies, systems, and products mentioned in this manual may not necessarily be indicated with a trademark symbol (TM, ®).

本マニュアルに記載されている会社名、システム名、製品名などには必ずしも商標表示(TM、®)を付記しておりません。

### **Open Source Software Notice**

Some modules or files of this product are subject to the terms of the following licenses. : APACHE2.0, CDDL1.0, EDL1.0, OPEN SYMPHONY SOFTWARE1.1, TRILEAD-SSH2, Bouncy Castle, BSD, MIT, SIL OPEN FONT1.1

Detailed Information related to the license can be found in the following directory : \${INSTALL\_PATH}/lib/licenses

この製品の一部ファイルまたはモジュールは、APACHE2.0、CDDL1.0、EDL1.0、OPEN SYMPHONY SOFTWARE1.1、TRILEAD-SSH2、Bouncy Castle、BSD、MIT、SIL OPEN FONT1.1のライセンスに準拠します。

### **文書情報**

文書名: JEUS セッション管理ガイド

発行日: 2016年10月14日

ソフトウェアバージョン: JEUS v8.0

ガイドバージョン: v2.1.1

---



# 目次

このガイドについて .....	ix
<b>第1章 セッション・トラッキング .....</b>	<b>1</b>
1.1. 概要 .....	1
1.2. セッション・トラッキングの構造 .....	1
1.3. セッション・トラッキングの動作 .....	3
1.3.1. Webエンジンでの動作 .....	3
1.3.2. クラスター環境での動作 .....	5
1.4. セッション・トラッキング・モード .....	13
1.5. コンテキスト間のセッション共有 .....	13
1.6. セッション・トラッキングの設定 .....	14
1.6.1. セッションの設定 .....	14
1.6.2. セッション・サーバーの設定 .....	20
1.7. セッション・トラッキングのチューニング .....	21
1.8. モニタリング .....	21
<b>第2章 分散セッション・サーバー .....</b>	<b>23</b>
2.1. 概要 .....	23
2.2. 基本概念 .....	23
2.3. サーバーの構造 .....	24
2.4. 動作方式 .....	27
2.5. 重複ログイン防止機能 .....	31
2.6. フェイルバック機能 .....	32
2.7. 同時要求セッションの維持機能 .....	34
2.8. セッション・クラスター・モード .....	36
2.8.1. デフォルト・セッション・クラスター・モード .....	37
2.8.2. ドメイン・スコープのセッション・クラスター・モード .....	37
2.8.3. 特別定義スコープのセッション・クラスター・モード .....	38
2.9. セッション・クラスターの基本設定 .....	39
2.10. セッション・クラスター・モード別の設定 .....	42
2.10.1. セッション・クラスター・モード別のクラスタリング参加方式 .....	42
2.10.2. 分散型セッション・サーバーの設定位置 .....	45
2.10.3. 分散型セッション・サーバーの設定 .....	48
<b>索引 .....</b>	<b>55</b>



# 図目次

[図 1.1]	JEUS Webエンジン構造のセッション関連部分 .....	2
[図 1.2]	Webエンジンのセッション・クッキーの発行手順 .....	4
[図 1.3]	セッションIDクッキーを使用してWebエンジンに2回目の要求を送信する手順 .....	4
[図 1.4]	セッションIDクッキーを使用して2つのWebエンジンとセッションを開始する場合 .....	6
[図 1.5]	スティッキー・セッション・ルーティンが存在しない場合はクライアント・セッションを削除 .....	7
[図 1.6]	セッション・クッキーにWebエンジンIDを追加付与 .....	8
[図 1.7]	スティッキー・セッション・ルーティングの動作プロセス .....	9
[図 1.8]	障害が発生したWebエンジンのトラブル .....	10
[図 1.9]	セッション・サーバーを使用する場合、クライアントの最初HTTP要求を処理する方法 .....	10
[図 1.10]	セッション・データ要求処理 .....	11
[図 1.11]	セッションの設定 - 基本情報 .....	15
[図 1.12]	セッションの設定 - 詳細設定 .....	17
[図 1.13]	セッションの設定 - 結果確認 .....	20
[図 2.1]	分散セッション・サーバーを利用したセッション・クラスタリングの構造 .....	24
[図 2.2]	分散セッション・サーバーの内部構造 .....	25
[図 2.3]	分散セッション・サーバーによるファイルオーバー・プロセス .....	29
[図 2.4]	重複ログイン防止機能の動作プロセス .....	31
[図 2.5]	フェイルバックをサポートしない場合 .....	33
[図 2.6]	フェイルバックをサポートする場合 .....	33
[図 2.7]	セッションの消失が発生する状況 .....	35
[図 2.8]	マイグレーションを実行しない場合の動作 .....	36
[図 2.9]	特別定義スコープのセッション・クラスターの構成例 .....	38
[図 2.10]	セッション・クラスタリングの設定 - 基本設定 .....	40
[図 2.11]	クラスタリング - クラスター・リストの照会 .....	47
[図 2.12]	クラスタリング - クラスターを構成するサーバーの選択および確認 .....	43
[図 2.13]	クラスタリング - 結果の確認 .....	43
[図 2.14]	セッション・クラスター・モード : DOMAIN_WIDE .....	44
[図 2.15]	セッションの選択 - Session画面 .....	46
[図 2.16]	セッション・クラスタリングの設定 - セッション・クラスターの追加 .....	45
[図 2.17]	セッションの選択 - Session画面 .....	46
[図 2.18]	分散型セッション・サーバーの設定 - 共通設定領域 .....	46
[図 2.19]	クラスタリング - クラスター・リストの照会 .....	47
[図 2.20]	分散型セッション・サーバーの設定 - サーバー・クラスターの設定位置 .....	49
[図 2.21]	Specific Scope Cluster - クラスター・リストの照会 .....	48
[図 2.22]	分散型セッション・サーバーの設定 - 特別定義スコープの設定位置 .....	49
[図 2.23]	分散型セッション・サーバーの設定 .....	49





# このガイドについて

## 対象読者

本書は、JEUS<sup>®</sup>(以下、JEUS) Webエンジンで使用されるセッション・マネージャーおよびセッション・サーバーの構成とその設定について記述したものです。クラスタリング環境で、あるいは単一サーバー内でセッションの保持および共有などを管理するシステム管理者と開発者を対象としています。

## 前提知識

本書を理解するには、以下の事項をあらかじめ熟知している必要があります。

- セッションについての基本的な知識
- HTTPセッションについての実務的な知識

JEUSの基本的な使用方法と製品を理解するには、以下のガイドについてあらかじめ熟知することをお勧めします。

- JEUS 紹介ガイド
- JEUS インストール & スタートガイド
- 『JEUS Webエンジンガイド』の「第1章 Webエンジン」

(セッション・マネージャーおよびセッション・サーバーの運用に関する情報)

本書のすべてのサンプルと環境構成は、UNIXスタイルに準拠します。Microsoft Windows<sup>™</sup>(以下、Windows)など他の環境で作業を行う場合は、次のような事項を考慮してください。

たとえば、Windowsプラットフォームでは、ディレクトリー区切り子をUNIXスタイルのスラッシュ(/)からWindowsスタイルのバックスラッシュ(\)に変えて使用してください。また、環境変数もWindowsスタイル(%%)に変更して使用してください。

本書で触れているJEUS\_HOMEは、JEUSがインストールされているディレクトリーです。

## 制限事項

本書の内容は、Java標準に準拠して作成されていますが、本書で触れているJava EEやJava仕様については詳しく取り上げていません。関連内容についてはJava関連ドキュメントを参照してください。

---

## 参考

Java EE 7仕様、Servlet 3.1仕様、JSP 2.2仕様などのJava EEに関連する内容は、  
<http://www.oracle.com/technetwork/java/index.html>を参照してください。

---

## 本書の構成

本書は、計2章で構成されています。

- 「第1章 セッション・トラッキング」

セッション・トラッキングの構造、動作、設定方法およびチューニングについて説明します。

- 「第2章 分散セッション・サーバー」

クラスタリング環境でセッション・トラッキングのために運用される分散セッション・サーバーの構造、動作および設定方法について説明します。

## 表記上の規則

表記	意味
<<AaBbCc123>>	プログラム・ソースコードのファイル名
<Ctrl>+C	CtrlキーとCキーを同時に押す
[Button]	GUIのボタン、メニュー名
太字	強調
「」、『』（鍵カッコ）	関連文書、あるいはガイド内の他の章および節の表示
「入力項目」	画面UI上の入力項目
ハイパーリンク	メール・アカウント、Webサイト
>	メニューの実行順
+----	下位ディレクトリー/ファイル有り
----	下位ディレクトリー/ファイル無し
<div>参考</div>	参照/注意事項
<div>注</div>	注意事項
[図 1.1]	図の名前
[表 1.1]	表の名前
AaBbCc123	Javaコード、XMLドキュメント
[ <i>command argument</i> ]	オプション・パラメータ
< xyz >	「<」と「>」の間の内容は実際に使用される特定の名前または値で置き換えられる
	構文の中の相互に排他的な選択項目の選択肢を示す 例) A B: AとBのいずれかを選択
...	パラメータ、値、または他の情報が繰り返される
\${ }	環境変数

## システム要件

	要求事項
プラットフォーム	Solaris 9, 10, 11
	HP-UX 11.x, 11i, 11iV2
	IBM AIX 5L, 6L, AIX 7L
	MS Windows 2008, 2012, Vista, 7, 8
ハードウェア	最小2GB以上、推奨20GBのハードディスク容量
	推奨1GB以上のメモリー容量
JDK	JDK 7, JDK 8

## 関連文書

ガイド	説明
JEUS 紹介ガイド	JEUSサーバーについて全般的に紹介し、JEUSのアーキテクチャーを含む各構成要素について記述しています
JEUS インストール&スタートガイド	JEUSについて紹介し、JEUSのインストールおよび開始方法について記述しています
JEUS サーバガイド	JEUSシステムおよびサーバーの概要とシステムの管理方法について記述しています
JEUS Webエンジンガイド	JEUS Webエンジンの管理方法、Java EE WARアーカイブとサーブレット/JSPの管理およびデプロイ方法について記述しています
JEUS EJBガイド	JEUS EJBエンジンおよびEJBモジュールのデプロイについて記述しています
JEUS リファレンスガイド	JEUSを使用するために必要な詳細設定とJEUSの使用方法について記述しています
JEUS WebAdminガイド	JEUSのWeb管理ツールであるWebAdminを利用したJEUSの設定および制御、モニタリング、クラスタリング、リソースの設定および管理について記述しています

## 参考文献

- Java EE 7標準
- Servlet 3.1標準
- JSP 2.2標準
- XMLリファレンスガイド

## お問合せ先

### Korea

TmaxSoft Co., Ltd.  
45, Jeongjail-ro, Bundang-gu,  
Seongnam-si, Gyeonggi-do, 13613  
South Korea  
Tel: +82-31-8018-1000  
Fax: +82-31-8018-1115  
Email: [info@tmax.co.kr](mailto:info@tmax.co.kr)  
Web (Korean): <http://www.tmaxsoft.com>  
TechNet: <http://technet.tmaxsoft.com>

### USA

TmaxSoft Inc.  
101 North Wacker Drive, Suite 2014,  
Chicago, IL 60606  
U.S.A  
Tel: +1-312-525-8330  
Email: [info@tmaxsoft.com](mailto:info@tmaxsoft.com)  
Web (English): [http://www.tmaxsoft.com/us\\_en/home](http://www.tmaxsoft.com/us_en/home)

### Japan

TmaxSoft Japan Co., Ltd.  
5F Sanko Bldg, 3-12-16 Mita, Minato-Ku, Tokyo, 108-0073  
Japan  
Tel: +81-3-5765-2550  
Fax: +81-3-5765-2567  
Email: [info@tmaxsoft.co.jp](mailto:info@tmaxsoft.co.jp)  
Web (Japanese): <http://www.tmaxsoft.co.jp>

## China

Beijing TmaxSoft System Software Co., Ltd.  
Room103, No.2 Huizhong Building, Seven Street Shangdi,  
Haidian District, Beijing, 100085  
P.R.China  
Tel: +86-10-6298-8827  
Email: [info@tmaxsoft.com.cn](mailto:info@tmaxsoft.com.cn)  
Web (Chinese): [http://www.tmaxsoft.com/cn\\_en/home\\_cn\\_en](http://www.tmaxsoft.com/cn_en/home_cn_en)

## Brazil

Tmax Brasil Sistemas e Serviços Ltda.  
Av. Copacabana, 177, sala 32~35 Empresarial 18 do Fortel  
Alphaville Barueri, Sao Paulo, 06472-001  
Brazil  
Tel: +55-11-4191-3100  
Fax: +55(11) 4191-3705 (extension#112)  
Email: [info.bra@tmaxsoft.com](mailto:info.bra@tmaxsoft.com)  
Web (Portuguese): [http://www.tmaxsoft.com/br\\_en/home\\_br\\_en](http://www.tmaxsoft.com/br_en/home_br_en)

## Russia

Tmax Rus L.L.C.  
Leninsky prospekt, 113/1 (Park Place Moscow),  
Office 318e, Moscow, 117198  
Russia  
Tel: +7(495)970-01-35  
Email: [info.rus@tmaxsoft.com](mailto:info.rus@tmaxsoft.com)  
Web (Russian): [http://www.tmaxsoft.com/ru\\_ru/home\\_ru\\_ru](http://www.tmaxsoft.com/ru_ru/home_ru_ru)

## Singapore

Tmax Singapore Pte. Ltd.  
430 Lorong 6, Toa Payoh #10-02,  
OrangeTee Building, 319402  
Singapore  
Tel: +65-6259-7223  
Fax: +65-6258-7112  
Email: [info.sg@tmaxsoft.com](mailto:info.sg@tmaxsoft.com)  
Web (English): [http://www.tmaxsoft.com/sg\\_en/home\\_sg\\_en](http://www.tmaxsoft.com/sg_en/home_sg_en)

## United Kingdom

TmaxSoft UK Ltd.  
215 Knyvett House, Watermans Business Park,  
The Causeway, Staines TW18 3BAB  
United Kingdom  
Tel: +44-1784-895005  
Email: [info.uk@tmaxsoft.com](mailto:info.uk@tmaxsoft.com)  
Web (English): [http://www.tmaxsoft.com/gb\\_en/home\\_gb\\_en](http://www.tmaxsoft.com/gb_en/home_gb_en)

## Canada

TmaxSoft Canada, Inc.  
2425 Matheson Blvd East, 8th floor,  
Unit 824 Mississauga, ON, L4W 5K4  
Canada  
Tel: +1-905-361-2888  
Email: [info.canada@tmaxsoft.com](mailto:info.canada@tmaxsoft.com)  
Web (English): [http://www.tmaxsoft.com/ca\\_en/home\\_ca\\_en](http://www.tmaxsoft.com/ca_en/home_ca_en)



## Australia

TmaxSoft Proprietary Limited  
L32, 101 Miller Street, North Sydney 2060  
Australia  
Tel: +91-9845-330-704  
Email: [info.aus@tmaxsoft.com](mailto:info.aus@tmaxsoft.com)  
Web (English): [http://www.tmaxsoft.com/au\\_en/home\\_au\\_en](http://www.tmaxsoft.com/au_en/home_au_en)

## India

TmaxSoft Technologies Private Limited  
Sobha Alexander Plaza, 3rd Floor,  
16/2 Commissariat Road, Bangalore-560025  
India  
Tel: +91-9845-330-704  
Email: [info.india@tmaxsoft.com](mailto:info.india@tmaxsoft.com)  
Web (English): [http://www.tmaxsoft.com/in\\_en/home\\_in\\_en](http://www.tmaxsoft.com/in_en/home_in_en)

## Turkey

TmaxSoft Co., Ltd. Turkey Liaison Office  
Windowist Tower. Eski Buyukdere Cad. No:26,  
Maslak 34467 Istanbul  
Turkey  
Tel: +90-544-553-6045  
Email: [cslee@tmaxsoft.com](mailto:cslee@tmaxsoft.com)  
Web (English): [http://www.tmaxsoft.com/tr\\_en/home\\_tr\\_en](http://www.tmaxsoft.com/tr_en/home_tr_en)



# 第1章 セッション・トラッキング

本章では、セッション・トラッキングの基本的な概念およびセッション、セッションID、セッション・クッキー、URLリライティングなどの定義について説明します。なお、JEUS Webエンジンと、より複雑な環境であるサーバー・クラスタリング環境でのセッション・トラッキングの実装や設定方法について説明します。

## 1.1. 概要

セッション・トラッキング(Session Tracking)とは、要求されたセッションを検索する動作です。クラスタリング環境においてセッション・トラッキングの範囲に従って、ルーティングによるセッション・トラッキングとセッション・サーバーによるセッション・トラッキングに分けられます。

---

### 参考

JEUS 6までは、中央集中型セッション・サーバーと分散型セッション・サーバーが運用されましたが、JEUS 7以降からは分散型セッション・サーバーのみ運用されます。

---

## 1.2. セッション・トラッキングの構造

本節では、基本的なセッション・トラッキングの構造について説明します。

---

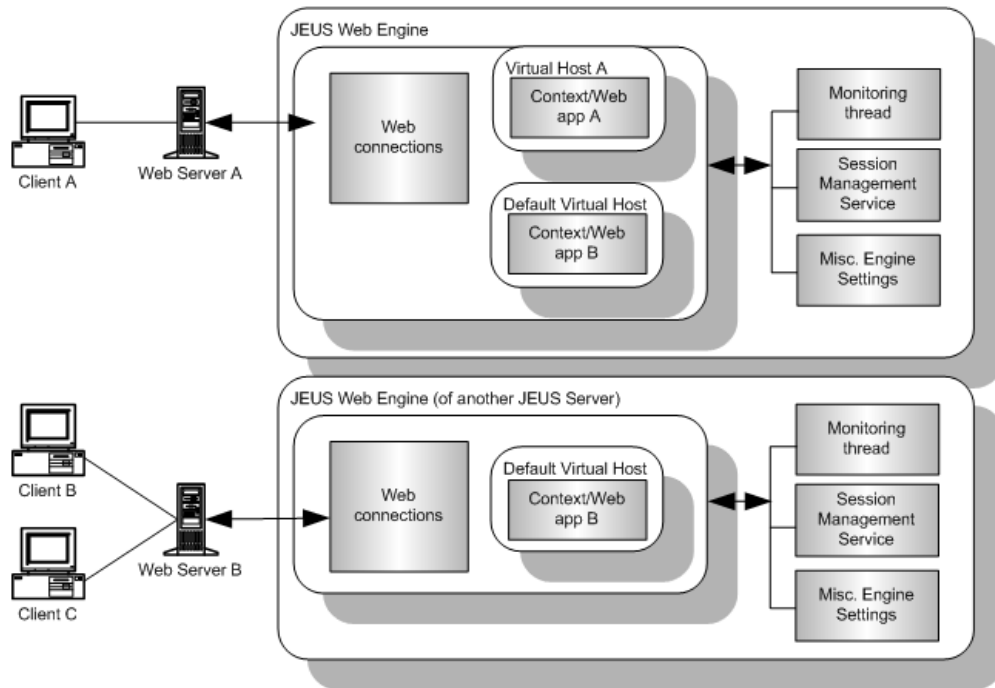
### 参考

本節では簡単に説明するので、サーブレットおよびセッション・トラッキングについての詳しい説明については、関連文書を参照してください。

---

以下の図は、セッション・トラッキングおよびセッション管理に関連するWebエンジンのコンポーネントを表しています。

**[図 1.1] JEUS Webエンジン構造のセッション関連部分**



HTTPセッションは、同じクライアント(Webブラウザなど)のHTTP要求に関連する一連の作業です。複数のクライアントがこのような要求を標準HTTPを使用して送信するとき、基本的にHTTPヘッダーには一意の「クライアントID」が存在しないため、Webサーバーはクライアントが区別できない問題が発生します。

したがって、Webサーバーはユーザーの要求を1つのセッションとしてトラッキングできません。これは、HTTPがステートレスかつコネクションレス・プロトコルであるためです。

---

#### 参考

Webサーバーもセッション・トラッキングに関連しています。

---

HTTP要求は以下のように動作します。

1. クライアントがWebサーバーに接続します。
2. クライアントがステートレスHTTP要求を生成します。
3. クライアントは応答を受信します。
4. HTTP接続が切断されます。

「クライアントID」や永続的なセッションの概念はHTTPプロトコルに含まれていません。したがって、WebサーバーはHTTP接続が切断されるか、要求に対する応答受信を終了した瞬間、各要求に関する事項を失います(単一要求の場合に発生)。そのため、複雑なWebアプリケーションでユーザーが持続的に相互関連する要求を実行する場合には使用できません。

この問題を克服するために、セッションIDという特殊なストリングを各HTTP要求に追加します。この一意のIDは、クライアントの最初要求時に必要に応じて生成され、クライアントに渡されます。以降、クライアントの要求に同セッションIDが持続的に付けられます。Webエンジンはこのような手順を通じて各要求の元が把握できるため、ユーザーがやり取りをしている間に会話型ステート(Conversational state)が保持でき、セッション機能のないHTTPプロトコルを利用してセッションの概念をサポートすることができます。

セッションIDは、クッキーまたはクライアントに返されるHTMLページの各URLリンクのパラメータとして自動的に追加されます。これをURLリライティングといいます。また別のオプションは、非表示フィールドとしてHTMLフォームにセッションIDを保存する方法があります。これらは、強力なサブリットAPIによって実装されるものです。

## 1.3. セッション・トラッキングの動作

本節では、JEUS Webエンジンおよびクラスタリング環境でのセッション・トラッキングの動作方式について説明します。

### 1.3.1. Webエンジンでの動作

JEUS Webエンジンはセッション・トラッキングを可能にするために、URLリライティングとクッキーをサポートしていますが、基本的にクッキーが使用されています。セッションIDを移動させるときに使われるクッキーをセッション・クッキーといいます。

Webエンジンにおいて、1つのセッションは1つのサブリットAPIであるHTTPセッション・クラスのインスタンスとして表現されます。同インスタンスは、セッション・クッキー(または、URLリライティングの結果として生成されたURLパラメータ)のセッションIDと関連付けられています。HTTPセッション・オブジェクトは、生成される場所であるWebエンジンに存在します。さらに、ユーザーの好みやオンライン・ショッピングなどの購入リストのようなユーザーに関するデータを持っています。

---

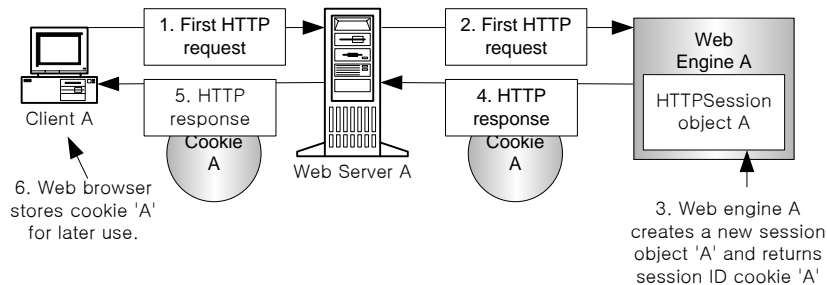
#### 参考

URLリライティングはセッション・クッキーの概念と類似しています。ただし、セッション・クッキーは別途のクッキー・ヘッダーに含まれますが、セッションIDはHTMLページのURLリンクに含まれる点が異なります。

---

セッション・クッキーは、1つのクライアントがJEUSのWebエンジンが管理するリソースを要求する際に使用されます。以下は、Webエンジンがセッション・クッキーを発行する手順です。

**【図 1.2】 Webエンジンのセッション・クッキーの発行手順**

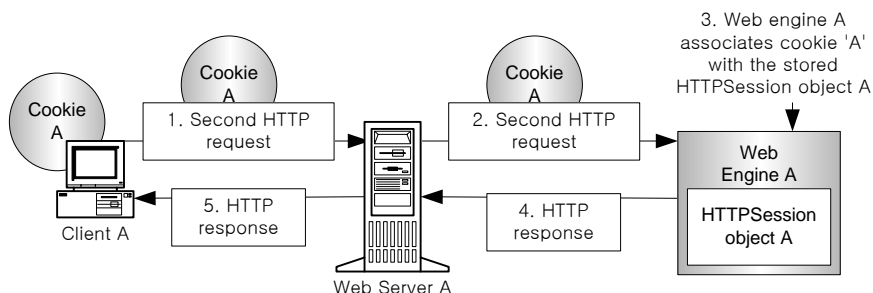


1. クライアントがWebサーバーに最初の要求をします。
2. WebサーバーはWebエンジンに要求を転送します。
3. Webエンジンは、クライアントのためのHTTPセッション・オブジェクトとHTTPセッションのセッションIDを持つセッション・クッキーを生成します。このように生成されたセッションIDは、同じクライアントが継続して要求するとき、メモリーで生成したHTTPセッション・オブジェクトを取得するために使われます。
4. 応答データとセッション・クッキーがWebサーバーに渡されます。
5. クライアントのWebブラウザーにセッション・クッキーと応答と一緒に渡され、HTTP接続は切断されます。
6. セッションIDを含むセッション・クッキーは、クライアントのWebブラウザーに保存されます。

上記の作業によって、クライアントはセッション・クッキーを持つことになり、同じWebエンジンに別の要求を送るときにクッキーと一緒に送信することができます。Webエンジンは、クッキーのセッションIDを使用してクライアントを認識し、クライアントのHTTPセッション・オブジェクトを取得することができます。

以下は、セッションIDクッキーを使って2回目の要求を送信する手順です。

**【図 1.3】 セッションIDクッキーを使用してWebエンジンに2回目の要求を送信する手順**



1. クライアントは同じWebサーバーに別の要求を送ります。その際、以前のセッション・クッキーをWebブラウザから取得して要求に添付します。
2. Webサーバーは、セッション・クッキーが含まれている要求を受け最初の要求と同様、同じWebエンジンに渡します。
3. Webエンジンは、要求とセッション・クッキーと一緒に受け取ります。セッション・クッキーから見つけたセッションIDに該当するHTTPセッション・オブジェクトを自身のメモリー領域からルックアップします。WebエンジンはルックアップしたHTTPセッション・データを使用して要求を処理します。
4. 応答データとセッション・クッキーがWebサーバーに渡されます。
5. HTTP応答がWebブラウザに渡されたら、HTTP接続は切断されます。セッション・クッキーは、最初の接続時にのみ応答に含まれて送信されます。以降は、応答にクッキーが含まれる必要はありません。

## 1.3.2. クラスター環境での動作

「1.3.1. Webエンジンでの動作」では、クライアント、Webサーバー、Webエンジンが1つずつ連携された簡単な状況におけるセッション・トラッキングについて説明しました。しかし、実際の運用環境では、このような簡単な構造では不十分な場合が多いです。多数のユーザー要求を受用するには、負荷分散とWebサーバー・クラスタリングが実現される必要があります。クラスター構成の詳しい内容については、『JEUS Webエンジンガイド』の「2.4. 負荷分散のためのWebサーバーの設定」を参照してください。

Webサーバー・クラスターでセッション・トラッキングのメカニズムを構成、設定するときは特別な注意が必要です。

以下は、分散されたクラスターでセッションを管理するときの主要事項です。

1. セッション・クッキーが含まれた要求を、最初に要求したWebエンジンに送信する方法
2. セッションを利用してすべてのWebエンジンが制限的な要求を処理するために、1つのエンジンで生成されたHTTPセッション・オブジェクトを別のWebエンジンで使用方法
3. 内部または外部の障害によってWebエンジンがダウンした場合、セッション・データをバックアップする方法

第1項の課題は、**スティッキー・セッション・ルーティング(Sticky Session Routing)**という機能で対処し、第2、3項の課題は、**セッション・サーバー(Session Server)**で対処できます。本節では、上記の3つの課題と解決方法について詳しく説明します。

---

## 参考

上記の第1項と2項の論点は根本的には同様ですが、スティッキー・セッション・ルーティングとセッション・サーバーという異なる方法で解決します。

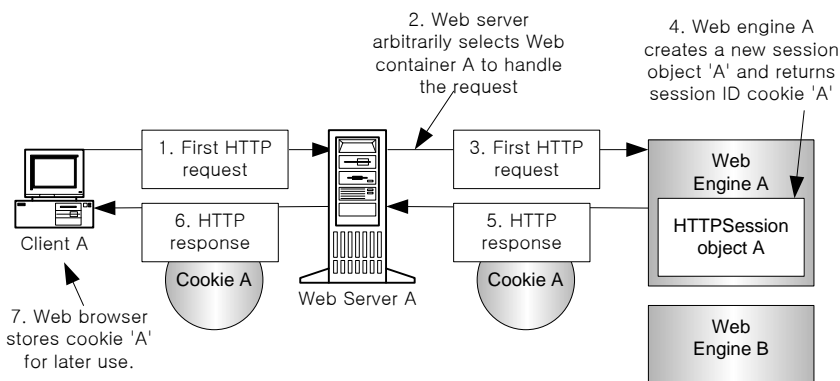
---

## スティッキー・セッション・ルーティング(Sticky Session Routing)

セッション・ルーティングとは、クラスターされた環境において、自身が生成または保存されたエンジンのIDをセッションに付与する機能です。エンジンIDがクッキーに含まれているため、Webサーバーでは該当するエンジンIDを確認し、要求することができます。それにより、Webエンジンのメモリーに保存されているセッションの効率を高められます。

たとえば、2つのWebエンジンA、Bが1つのWebサーバーに接続されたクライアント要求状況を以下のように表すことができます。

**[図 1.4] セッションIDクッキーを使用して2つのWebエンジンとセッションを開始する場合**

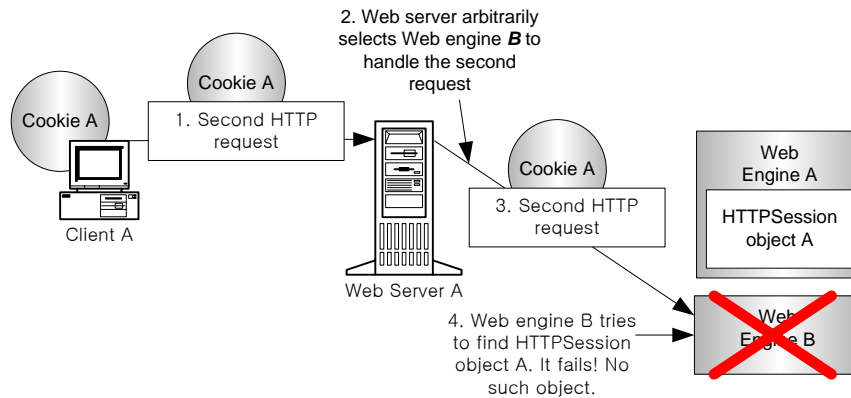


1. クライアントがWebサーバーに最初の要求を送ります。
2. Webサーバーは要求を送信するため、1つのWebエンジンを選択します。上記の例では、WebエンジンAが選択されています。
3. 要求がWebエンジンAに送信されます。
4. WebエンジンAはHTTPセッション・オブジェクトを生成し、応答とセッションIDクッキーと一緒に返します。このIDは、次回の同じクライアントの要求を処理するHTTPセッション・オブジェクトをメモリーから呼び出すときに使用されます。
5. WebエンジンはWebサーバーに応答とセッション・クッキーを返します。
6. クライアントのWebブラウザにセッション・クッキーと応答が渡され、HTTP接続は切断されます。



最初の要求は問題なく正常に終了しました。しかし、2回目の要求が同じクライアントから生成されるときは深刻な問題が発生します。

**[図 1.5] スティッキー・セッション・ルーティンが存在しない場合はクライアント・セッションを削除**

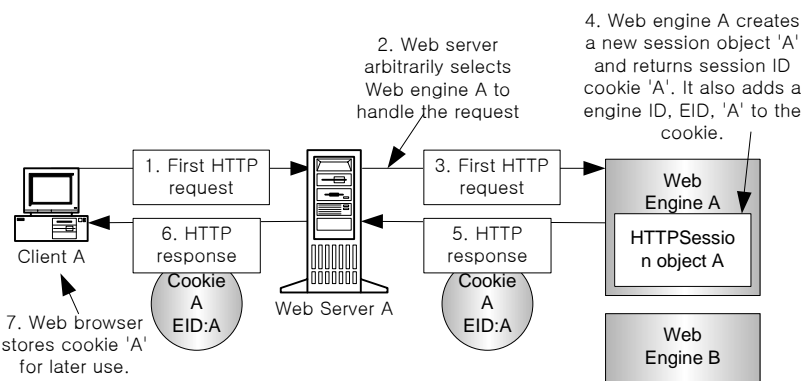


1. クライアントが同じWebサーバーに別の要求をします。最初の要求で返されたセッション・クッキーをWebブラウザから取得し、要求に付けます。
2. Webサーバーは要求を受け、2つのWebエンジンから任意で1つのWebエンジンを選択します。本例ではWebエンジンBが選択されました。
3. 要求とセッション・クッキーがWebエンジンBに送信されます。
4. WebエンジンBは要求とセッション・クッキーを受け取ります。自身のメモリー領域からクッキーに対応するHTTPセッションを取得しようとしませんが、対応するHTTPセッション・オブジェクトが存在しないため取得できません。そのため、WebエンジンBはクライアント・セッションが維持できず、新しいセッションを強制的に生成するか、エラー・メッセージを返します。(これらのオプションは推奨しません。)

上記の説明のようにエンジン数を増やし、同じサービスを実行してトラフィックを分散させることはできます。しかし、実際に持っている情報が見つからないか、転送できないため問題が発生します。

同問題を解決するため、最初にHTTPセッション・オブジェクトを生成した同じWebエンジンのセッションで、制限された要求を正常にルーティングできるようにセッション・クッキーにWebエンジンIDを追加で付与します。以下の図は解決手順です。

**[図 1.6] セッション・クッキーにWebエンジンIDを追加付与**



1. クライアントがWebサーバーに最初の要求をします。
2. Webサーバーは任意のWebエンジンを選択します。本例では、WebエンジンAが選択されています。
3. 要求がWebエンジンAに送信されます。
4. WebエンジンAはHTTPセッション、セッションIDクッキーを生成し、WebエンジンID(EID)をクッキーに追加します。
5. Webサーバーに応答とセッション・クッキーを返します。
6. Webブラウザーに応答とセッション・クッキーを返します。
7. WebエンジンID(EID)を含むセッション・クッキーがWebブラウザーに保存されます。

---

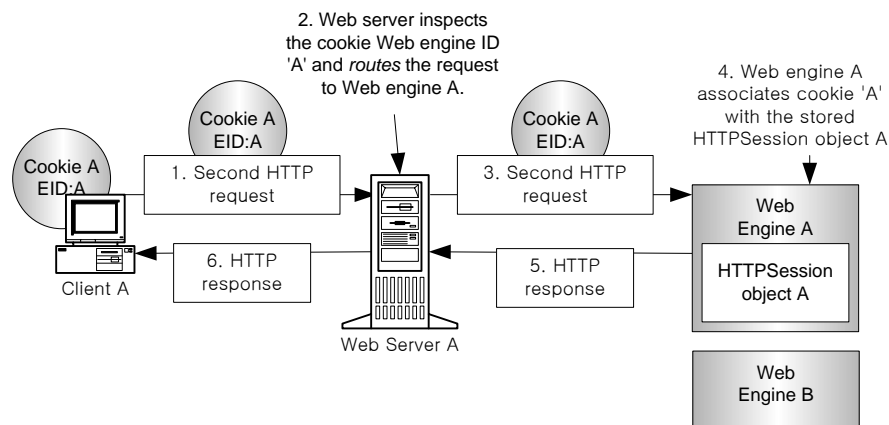
#### 参考

JEUS 6ではエンジンのIDを直接入力して運用しましたが、JEUS 8では該当する情報をエンコーディングして運用します。

---

以下は、スティッキー・セッション・ルーティングの動作プロセスです。

【図 1.7】スティッキー・セッション・ルーティングの動作プロセス



Webサーバーは2回目の要求に対して、セッション・クッキーとエンジンID値(EID)を検索します。Webサーバーは該当するHTTPセッションが存在する元のWebエンジンに要求をルーティングします。

#### 参考

Webサーバーで標準ではないエンジンIDを識別するため、Apache、IIS、SunOne(Iplanet)のような別のWebサーバーの場合は、mod\_jkモジュールをインストールする必要があります。WebtoB Webサーバーには同機能が組み込まれています。mod\_jkモジュールのインストールについては、『JEUS Webエンジンガイド』の「2.4. 負荷分散のためのWebサーバーの設定」を参照してください。

スティッキー・セッション・ルーティング機能を使用するには、すべてのWebサーバーとWebエンジンが接続される必要があります。負荷分散機を使用する状況において、複数のWebサーバーのうち要求を受けたWebサーバーが該当するWebエンジンに接続できない場合は、接続が切断される可能性があります。しかし、負荷分散機がセッション・ルーティングをサポートしている場合は、すべてのWebサーバーとWebエンジンが接続される必要はありません。たとえば、WebtoBを負荷分散機として使用する場合、クラスタリングが完璧に接続されていなくてもセッション・ルーティングを使用することができます。

スティッキー・セッション・ルーティング機能は、別の動作には影響を与えず、要求を転送するルーティングにのみ関連しています。スティッキー・セッション・ルーティングが効率的に実行されれば、セッションの重複生成および重複保存を防ぐことができるため、性能が大幅に高まります。しかし、スティッキー・セッション・ルーティングを使用しても要求の送信に対して強制性はありません。スティッキー・セッション・ルーティングは効率を高める機能なので、要求を送信するエンジンに負荷がかかるか、障害状況の場合は別のエンジンに送られます。すなわち、スティッキー・セッション・ルーティング機能だけでは、完璧なクラスタリング環境は構成できません。

## セッション・サーバー

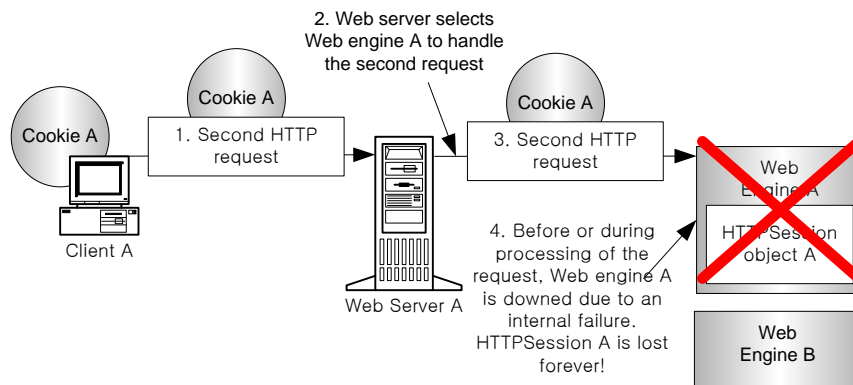
簡単なスティッキー・セッション・ルーティングに比べ層強力なのがセッション・サーバーです。

セッション・サーバーを使用する場合は、すべてのWebエンジンがクラスター内のすべてのWebサーバーに接続される必要はありません。また、セッション・サーバーはクラスター内のすべてのセッション・データに対

して信頼的なバックアップ機能を提供します。したがって、特定のWebエンジンに障害が発生してもセッション・データは保存され、別の正常なWebエンジンが代わりに要求を処理します。

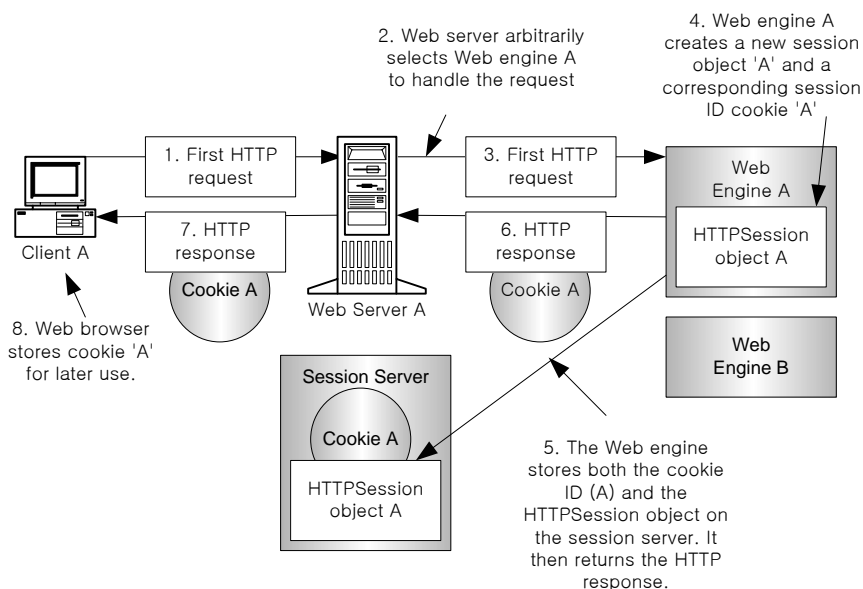
以下は、Webエンジンに存在するすべてのユーザーのセッション・データが消失されるトラブルが発生したWebエンジンの要求を処理する手順です。このような状況が運用環境で発生してはいけません。

**【図 1.8】 障害が発生したWebエンジンのトラブル**



このように特定のWebエンジンに障害が発生してもセッションを保持するために、セッション・サーバーがクラスターに追加されました。セッション・サーバーを使用する場合、クライアントの最初のHTTP要求が以下の方法で処理されます。

**【図 1.9】 セッション・サーバーを使用する場合、クライアントの最初HTTP要求を処理する方法**

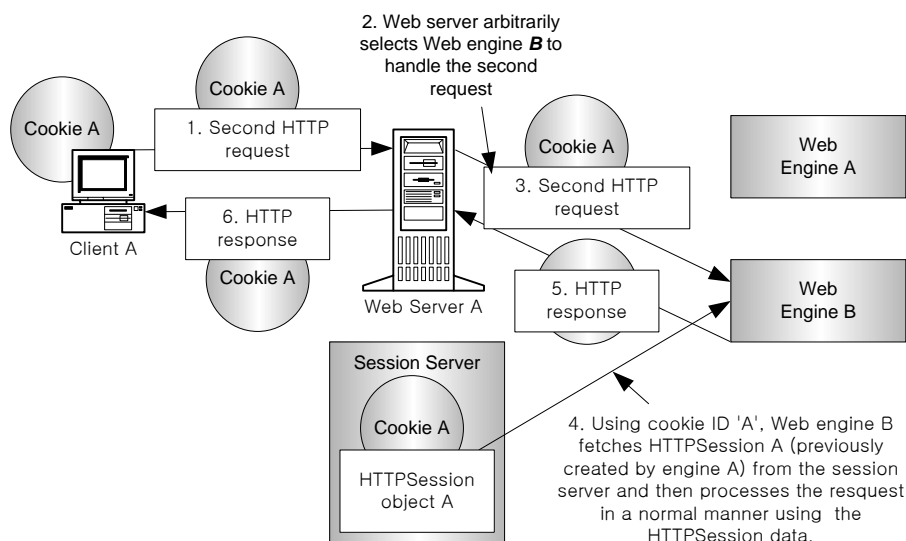


1. クライアントがWebサーバーに要求を送ります。
2. Webサーバーはクラスター内でWebエンジンAを選択して要求を処理します。
3. WebサーバーはWebエンジンAに要求を送信します。

4. WebエンジンはHTTPセッション・オブジェクトとセッション・クッキーを生成します。このIDは、次回同じクライアントから要求が送られる場合、生成されたHTTPセッション・オブジェクトをセッション・サーバーから取り出すために使われます。
5. 要求に対する処理が完了したら、WebエンジンはHTTPセッション・オブジェクトとセッションIDをセッション・サーバーに保存します。
6. 応答データとセッション・クッキーはWebサーバーに渡されます。
7. セッションIDクッキーと応答データがWebブラウザに渡され、HTTP接続が切断されます。
8. Webブラウザは送られたセッション・クッキーを保存します。

以降、WebサーバーがクライアントAの要求をWebエンジンBが処理するように指示したり、WebエンジンAに障害が発生しても、セッション・サーバーからセッション・データを取得することができます。

**[図 1.10] セッション・データ要求処理**



セッション・サーバーは上記の図のように共有リポジトリの役割をします。WebエンジンAとWebエンジンBはそれぞれ異なる保存媒体として、互いの内容が参照できないように構成されています。しかし、セッション・サーバーを両方の共有リポジトリとして構成し、セッション情報を保存したり呼び出したりしてセッションの共有をサポートします。

概念的にエンジンAとエンジンBを別途のスペースに、セッション・サーバーを共有されるスペースとして表現しています。しかし、WebエンジンA、WebエンジンBもセッション・サーバーのように共有されるスペースであることもあります。こちらでいう共有とは、ネットワーク、つまり別のインターフェースなどを介してアクセスできるという意味です。

より安定的にセッション・データを保存するため、障害状況に備えてバックアップ・セッション・サーバーを運用しています。バックアップ・セッション・サーバーは、内部的に運用されるサーバーから自動で選択され、サーバーの追加やダウンおよびフェイルされる場合にも自動アップデートされます。

既存のバージョンで提供していた中央型セッション・サーバーのように別途の保存スペースを設け、どこのWebエンジンからもアクセス可能なインターフェースを提供しているため、特定のWebエンジンに障害が発生してもセッションが保持できるメリットがあります。

しかし、中央型セッション・サーバーの方式は、すべてのWebエンジン・セッションが集中されるため、大規模のクラスタリング環境では性能が低下します。また、複数のサーバーが存在しているにも関わらず、1つのサーバーのみ動作する部分も適切ではありません。

この問題を解決すべく、基本的に分散型セッション・サーバーを運用しています。分散型セッション・サーバーは、大規模クラスタリング環境で性能を高めるために考案されたセッション・サーバーです。

分散型セッション・サーバーは、Webエンジンごとにセッション・サーバーを設ける方式です。基本的にセッション・ルーティング機能を使用しており、セッション・サーバーと同様、セッション・データのバックアップが設定できるため、Webエンジンに障害が発生しても継続的にセッションを保持することができます。

分散型セッション・サーバーは、サーバー・クラスターを構成すると自動で使用されます。サーバー・クラスターを使用する場合、特に設定がなくても、クラスターに参加したサーバーのコンテキストは分散型セッション・サーバーを構成してセッションを共有し維持します。分散型セッション・サーバーの詳細設定については、[「2.10.3. 分散型セッション・サーバーの設定」](#)を参照してください。

## 混合モード(Mixed Mode)

混合モードとは、スティッキー・セッション・ルーティングとセッション・サーバー方式を組み合わせたものです。各機能の長短所は以下のとおりです。

	スティッキー・セッション・ルーティング方式	セッション・サーバー方式
長所	Webエンジンのセッション・オブジェクトにアクセスするので速度が速いです	セッション・サーバーにすべてのセッション・オブジェクトが保存されるため、特定のWebエンジンに問題が発生してもセッション・オブジェクトが消失されず、セッションが保持できます
短所	トラブルが発生した場合、Webエンジンのすべてのセッション・データが消失され、復旧できません	要求が受信されたとき、セッション・サーバーからセッション・オブジェクトを取得する必要があり、HTTPセッション・オブジェクトが変更された場合は、セッション・サーバーに再保存しなければなりません。そのため、スティッキー・セッション・ルーティング方式より速度が遅いです

混合モードを使用すると2つの方式のメリットを活かすことができます。2つの方式を組み合わせると、セッション・オブジェクトはセッション・サーバーとセッション・オブジェクトを生成したWebエンジンに存在することになり、Webエンジンは必要な場合のみセッション・サーバーからHTTPセッション・オブジェクトを取り出して変更します。混合方式を使用すると、セッション・サーバーだけを使用する場合に比べ、使われるネットワーク

帯域幅が半分程度減り、すべてのセッション・データの安全なバックアップも保証できます。したがって、クラウド環境では混合方式を利用したセッション管理をお勧めします。

## 1.4. セッション・トラッキング・モード

サブレット3.1では、既存のセッション・トラッキングの様々な方法が選択できる機能を提供します。

- クッキー使用モード

既存の基本的なセッション・トラッキング・モードの1つとしてクッキーを利用します。

クッキーはブラウザの動作と密接な関係であり、ほとんどはこのようなクッキーを利用してセッションを保持、動作しています。しかし、クッキーもブラウザに依存的なので、ルールが異なるところでは使用できず、セッションが保持できない場合もあります。すなわち、クッキーのドメインとパスに関する基本的な知識が要求されます。このようなルールにより目的のセッションに対するトラッキングが可能となります。

- URLモード

URLリライティング・モードは、ブラウザがクッキーをサポートしておらず、クッキー使用モードが使えない場合のために存在するモードです。

JEUS 6のURLリライティングと同様な方法です。つまり、クッキーを使用せずに対象URLに目的のサーバーに関する情報を記録します。このような機能で、Webブラウザがクッキーをサポートしなくてもセッションを保持することができます。

このモードを使用するため、jeus-web-dd.xml DD(Deployment Descriptor)に特殊なタグが必要です。<tracking-mode><url>タグを使用すると、セッションIDはURLリライティングによって保持されます。このような方法で、セッション・トラッキングは別のドメイン名がいくつかの要求に使われても動作します。

- SSL使用モード

SSL使用モードは、SSL接続にセッションを制限的に転送したい場合に設定します。

JEUSを運用する際、セキュリティ上保護が必要なデータをセッションに格納する場合に設定するモードです。SSL使用モードを適用すると、SSL接続以外の場所からのセッションは転送されません。

## 1.5. コンテキスト間のセッション共有

通常、セッションは同じコンテキストでのみ管理されますが、別のコンテキスト間の共有も可能です。

セッション・サーバーを使用してもすべてのセッションが共有されるわけではなく、互いに異なるコンテキスト間のセッション・データを共有したい場合は、セッション設定の**Shared**を設定します。Shared設定はセッション・データを共有するために必ず設定します。詳しい設定については、[「1.6.1. セッションの設定」](#)を参照してください。

## 1.6. セッション・トラッキングの設定

セッション・トラッキングのためにスティッキー・セッション・ルーティング、またはセッション・サーバー機能を使用する場合は、以下の設定が必要です。

### ● スティッキー・セッション・ルーティングの設定

デフォルトでスティッキー・セッション・ルーティングをサポートしているため、別途設定は必要はありません。

エンジン情報のエンコーディングを追加で設定できます。

項目	説明
BASE64	スティッキーされる情報をBASE64のルールに従ってエンコーディングして転送します(デフォルト値)。  セッションIDによりエンジン情報やドメイン情報が露出しないように、意味のない情報に見せかけるためにエンコーディングします
RAW	スティッキーされる情報をエンコーディングせずにそのまま転送します。  エンコーディングされた情報は内部でもどのエンジンからの要求かが判断し難いため、デバッグする場合や、エンジン名の露出がセキュリティに影響を与えない場合に設定します

### ● セッション・サーバーの設定

分散型セッション・サーバーを使用するには、クラスター設定で参加させるサーバーをクラスターに追加します。

---

#### 参考

セッション・サーバーおよびスティッキー・セッション・ルーティングについての詳細は、「[2.9. セッション・クラスターの基本設定](#)」を参照してください。コンソール・ツール(jeusadmin)を使用して設定することも可能です。コンソール・ツールで使用する設定関連のコマンドについては、『JEUS リファレンスガイド』の「4.2.9.5. modify-session-configuration」を参照してください。

---

### 1.6.1. セッションの設定

セッションを管理するために、セッション・オブジェクトの共有有無、セッション・クッキーの設定、タイムアウトなど、Webエンジンのセッションと関連するすべての事項を設定します。

WebAdminを使用してセッションを設定する方法は以下のとおりです。



1. WebAdminの左側のメニューから[Servers]を選択すると、サーバー・リストの照会画面が表示されます。サーバー・リストから目的のサーバーを選択すると、サーバー設定画面に移動します。設定画面で、[Engine] > [Web Engine] > [Session Config]メニューを選択します。
2. [LOCK & EDIT]ボタンをクリックして設定変更モードに切り替えます。
3. [Session Config]画面は、以下のように**基本設定**と**詳細設定**に分けられます。同項目は、Webエンジンで共通的に使用するセッション設定を定義します。コンテキスト別に設定をオーバーライドすることが可能で、優先順位はコンテキスト、Webエンジンの順です。各項目を設定した後、[確認]ボタンをクリックします。

【図 1.11】セッションの設定 - 基本情報

Session Config

HISTORY

Webエンジンで共通して使用されるセッションを設定します。コンテキストごとに、この設定をオーバーライドできます。コンテキストの設定がWebエンジンの設定より優先されます。

Basic Resource Engine

Web Engine Jms Engine Ejb Engine

Basic Jsp Engine Virtual Host Web Connections Access Log Session Config

動的設定 \* 必須項目 このページの設定を変更したい場合は、左のメニューから[Lock & Edit]ボタンをクリックしてください。 TIP

Timeout	30 m [デフォルト: 30] サーバで作成されるセッションの有効期限を設定します。最後に使用されてから指定の時間が経過したセッションは削除されます。この設定は、サーブレットで設定するweb.xmlのセッションタイムアウト設定より優先順位が低いです。個別セッションはweb.xmlに設定できます。-1を設定した場合は、セッションを削除しません。セッションの有効期限を分単位で設定します。設定していない場合、デフォルト値は30分です。
Max Session Count	-1 [デフォルト: -1] メモリに維持する最大セッション数を設定します。デフォルト値は-1です。(無制限)セッション数が設定値に達した場合、セッションの作成要求があると、エラーが発生します。値を設定しない場合は、セッションを無制限に作成できます。
Shared	<input type="checkbox"/> [デフォルト: false] 1つのコンテキストで作成されたセッションオブジェクトを他のコンテキストでもアクセスできるように、コンテキスト間でセッションを共有するか否かを設定します。コンテキスト間のセッション共有を有効にしていない場合、セッションはアプリケーション単位で共有されます。コンテキスト間のセッション共有を有効にした場合、該当するサーバ内でセッションが共有されます。ブーリアン型で、デフォルト値はfalseです。

以下は、設定項目についての説明です。

項目	説明
Timeout	<p>作成されたセッション・オブジェクトの有効周期を設定します。</p> <p>通常、Webアプリケーション設定のweb.xmlでセッション・タイムアウトを設定しますが、web.xmlで設定しない場合は、Timeoutの設定値が適用されます。すなわち、web.xmlのセッション・タイムアウト設定が存在すると、Timeoutの設定値は適用されません。(web.xmlのセッション・タイムアウトの優先順位が一番高いです。)</p> <p>整数型の分単位で設定します。設定しないとデフォルト値が適用されます</p>

項目	説明
Max Session Count	<p>メモリーに維持するセッションの最大数を設定します。設定した数を超えるセッションの作成要求が入るとエラーを発生させます。</p> <p>特定のアプリケーションのセッション数の増加が他のアプリケーションのセッションの作成に影響を与えてはならないため、アプリケーションごとにこのセッション数が適用されます。</p> <p>デフォルト値は -1であり、無制限のセッションの作成を許可することを意味します</p>
Shared	<p>コンテキストで生成されたセッション・オブジェクトをコンテキスト同士が共有するかについて設定します。つまり、コンテキストAで生成されたHTTPセッション・オブジェクトがコンテキストBでも同じユーザーが使用できるかについての設定です。</p> <p>セッション・クラスタリングでない環境の場合、最大の共有領域は同じアプリケーション内のコンテキストです。セッション・クラスタリング環境の場合は、共有領域の制限はありません。</p> <p>設定しないとデフォルト値が適用されます</p>

## 参考

Sharedを指定し、セッションが複数のコンテキストで共有される場合、JEUS Webエンジンにおいてのセッション・タイムアウトはセッションが最初に生成されたコンテキストのTimeout設定に従います。

以下は、**Session Config**画面の**詳細設定**です。

[図 1.12] セッションの設定 - 詳細設定

[詳細設定](#)

[すべてを開く](#)

Session Config

Reload Persistent

☐

Tracking Mode

Cookie

☒

Url

☐

Ssl

☐

Session Cookie

Cookie Name

JSESSIONID

Version

0

Domain

Path

Max Age

-1

Secure

☐

Http Only

☒

Comment

確認

再設定

# - Session Config

項目	説明
Reload Persistent	<p>サーブレット・コンテキストが変更され、リロードが発生すると該当するコンテキスト内のセッション・オブジェクトの属性はすべて削除されます。設定しないとデフォルト値が適用されます</p> <ul style="list-style-type: none"> <li>– true : リロードを実行してもセッション・オブジェクトの属性は保持されます</li> <li>– false : リロードを実行すると、セッション・オブジェクトのすべての属性は削除されます(デフォルト値)</li> </ul>

## ● Tracking Mode

セッションを転送する方法(セッション・トラッキング)を設定します。以下は、セッション・トラッキングを設定する3つの方法で、重複設定が可能です。設定しない場合は、クッキーによるトラッキングのみ使用できるように設定されます。

項目	説明
Cookie	セッション・トラッキング方法として、クッキーを使用する場合の設定です – true : クッキーを利用したセッション・トラッキングを使用します(デフォルト値、推奨) – false : クッキーを利用したセッション・トラッキングを使用しません。セッション保持が正常にできない場合があります
Url	セッション・トラッキングの方法として、URLリライティング方式を使用する場合の設定です – true : URLリライティング方法を使用します – false : URLリライティング方法を使用しません(デフォルト値)
Ssl	セッション・トラッキング方法として、SSLを使用する場合の設定です – true : セッション・トラッキング時に、SSLを介してのみセッションを転送します。正常に動作しない場合があります – false : セッション・トラッキング時に、SSL接続以外でもセッションの転送が可能です (デフォルト値)

## ● Session Cookie

ユーザーのセッションを追跡する基本技術は、すべてのクライアント応答に返されるセッション・クッキーを利用して実装されます。

これは、Webエンジンから応答を送信する際のHTTPヘッダーのセッション・クッキーに対する設定です。一般的にはエンジンでクッキーを構成しますが、特別なクッキー情報を構成するときに使用します。

以下の詳細設定が可能です。

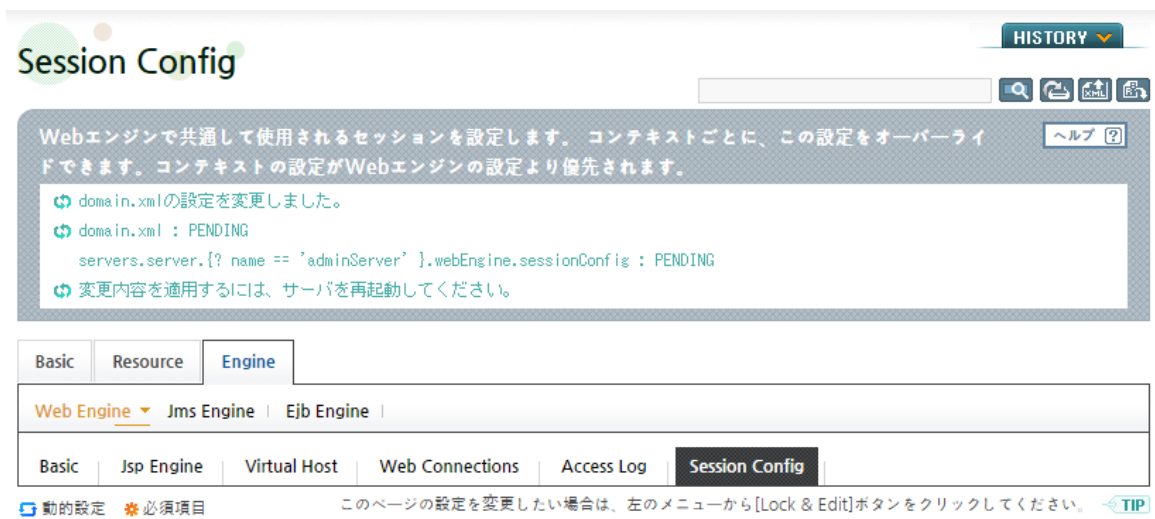
項目	説明
Cookie Name	セッション・クッキーの名前として標準名の「JSESSIONID」を使用せずに、別の名前を使用する場合に設定します。文字列型の設定で、設定しないとデフォルト値が適用されます(デフォルト値: JSESSIONID)

項目	説明
Version	<p>クッキーIDのバージョンを設定します。整数型の設定で、以下の値のうち1つを設定します</p> <ul style="list-style-type: none"> <li>– 0 : NSクッキー・タイプです(デフォルト値)</li> <li>– 1 : RFC仕様のクッキー・タイプです</li> </ul>
Domain	<p>セッション・クッキーが送られるサーバーのドメイン名を指定します。クッキーは、このドメイン要求に対してのみ返されます。ドメイン名は「.」で始まり、&lt;host_name&gt;を指定してはいけません。詳細内容については、「RFC-2109仕様」を参照してください。</p> <p>文字列型の設定です。設定しない場合は、クッキーにドメイン情報が含まれません</p>
Path	<p>セッション・クッキーが送られるドメイン内のUELパスを文字列型で指定します。ドメインが適合する場合、該当するURLにクッキーと要求と一緒に送られます。</p> <p>たとえば、「/examples」というパスが指定され、ドメインは「.foo.com」に設定されている場合、クライアントの要求は「www.foo.com/examples」である場合のみクッキーと一緒にサーバーへ送られます。詳細内容については、「RFC-2109仕様」を参照してください。</p> <ul style="list-style-type: none"> <li>– 設定しない場合 : エンジン内部から適切なパスを選択します</li> <li>– 設定した場合 : 設定された固定パス情報が常にクッキー情報に含まれます。</li> </ul> <p>パスのルート値である「/」ではなく、別の値で設定する場合は、アプリケーション・セッション共有の特性を考慮して慎重に設定します</p>
Max Age	<p>セッション・クッキーのexpires属性を設定します。指定時間を過ぎるとクッキーはクライアントから削除され、それ以上は送られません。整数型で設定し、設定しないとデフォルト値が適用されます。(デフォルト値: -1、単位: 秒)</p> <p>デフォルト値が設定されると、クッキーの「expires」属性を使用しないこととなります。つまり、ブラウザのライフサイクルに従うとの意味で、ブラウザがクローズされるとき、クッキーはユーザーのセッションと同時に終了します</p>
Secure	<p>セッション・クッキーのsecure属性をブーリアン型で設定します。設定しないとデフォルト値が適用されます</p> <ul style="list-style-type: none"> <li>– true : セッション・クッキーは、Secure http connectionであるHTTPSの場合のみ送られます</li> <li>– false : 別の接続でもセッション・クッキーが送られます(デフォルト値: false)</li> </ul>

項目	説明
Http Only	<p>セッションIDクッキーのHttpOnly属性をブーリアン型で設定します。設定しないとデフォルト値が適用されます。</p> <p>HttpOnly属性は、サーブレット3.0に追加された機能です。HTTP以外のスクリプト要求によってクッキーが使用されることを防ぐためのセキュリティ機能です</p> <ul style="list-style-type: none"> <li>– true : HTTP要求にのみセッション・クッキーが使用されます(デフォルト値)</li> <li>– false : スクリプト・コードなどでもセッション・クッキーが使用されます</li> </ul>
Comment	<p>ユーザーがクッキー情報を容易に把握できるように、目的または説明を記録します。</p> <p>Netscapeバージョン0のクッキーの場合はサポートされません</p>

4. 設定が完了したら、[**Activate Changes**]ボタンをクリックして変更内容を反映します。
5. 設定内容が反映されると、以下のようにその結果が画面に表示されます。[**Session Config**]画面の各項目は動的設定ではないので、変更内容を反映するためには表示されたメッセージのようにサーバーを再起動する必要があります。

**[図 1.13] セッションの設定 - 結果確認**



## 1.6.2. セッション・サーバーの設定

ドメイン構造でセッション・サーバーを使用するには、クラスタリングに参加する必要があります。WebAdminでクラスターに参加するサーバーを設定します。このような設定によって、コンテキスト別に設定しなくてもセッション・サーバーを使用することができます。クラスタリングの構成および分散セッション・サーバーの設定方

法については、それぞれ「[2.8. セッション・クラスター・モード](#)」と「[2.10.3. 分散型セッション・サーバーの設定](#)」を参照してください。

## 1.7. セッション・トラッキングのチューニング

クラスターされた環境で最適な性能を出すために以下のように実行します。

- 良好な性能かつ安定的な運用を保証するため、常にスティッキー・セッション・ルーティングとセッション・サーバーを混合して使用します。
- Webサーバーが正常に接続されてからチューニングされる必要があります。
- ユーザーの要求が集中するサイトでは、分散セッション・サーバーの使用をお勧めします。

## 1.8. モニタリング

生成されたセッションに対する基本的なモニタリングを通じて、現在のサーバーが持っているセッション数を確認することができます。

セッションのモニタリングには、以下の2つの方法があります。

- WebAdminを使用したモニタリング

WebAdminを使用してセッションをモニタリングする方法は、『*JEUS Webエンジンガイド*』の「1.4. 管理ツール」の「Webエンジン・モニタリング」を参照してください。

- コンソール・ツールを使用したモニタリング

コンソール・ツールを使用してセッションをモニタリングする方法は、『*JEUS リファレンスガイド*』の「4.2.8.34. show-web-statistics」と『*JEUS リファレンスガイド*』の「4.2.9.2. list-session」を参照してください。





# 第2章 分散セッション・サーバー

本章では、クラスタリング環境でセッション・トラッキングのために運用される分散セッション・サーバーの構造、動作および設定方法について説明します。

## 2.1. 概要

セッション・サーバーはクライアントのセッション・データを管理およびバックアップするために使用されます。特に、複数のWebサーバーとサーブレット・エンジンがクラスタリングされた環境でセッション・データを管理するときに有用です。分散型セッション・サーバーを運用します。

---

### 参考

JEUS 6までは主に中央型セッション・サーバーが使用されましたが、システムの中央集中によるトラブルを減らすために、JEUS 7以降からは分散型セッション・サーバーのみ使用されます。

---

## 2.2. 基本概念

分散セッション・サーバーは、セッション・クラスタリング機能を提供しつつ、拡張性および安定性を改善した方式です。したがって、大規模クラスタリング環境でその性能が発揮できます。

分散セッション・サーバーは、基本的にセッション・サーバーとしての動作面では既存の中央セッション・サーバーと同様です。複数のサーブレット・エンジンで構成されたクラスタリング環境で、同じクライアントからの要求が特定のサーブレット・エンジンで処理されなくてもセッションが保持できる機能を提供します。

分散型セッション・サーバーは、セッションの管理主体が分散されていることを意味しており、それによって高い拡張性を提供します。

分散セッション・サーバーの特性は以下のとおりです。

- 複数のサーブレット・エンジンで構成されたクラスタリング環境において継続的なセッションの維持が可能です。
- 直前の要求を処理していたサーブレット・エンジンがダウンしても、別のサーブレット・エンジンが以降の要求を処理する際にセッションが切断しないようにします。
- 分散型プロトコルを使用するため、クラスタリングの規模が大きくなっても拡張性が容易です。

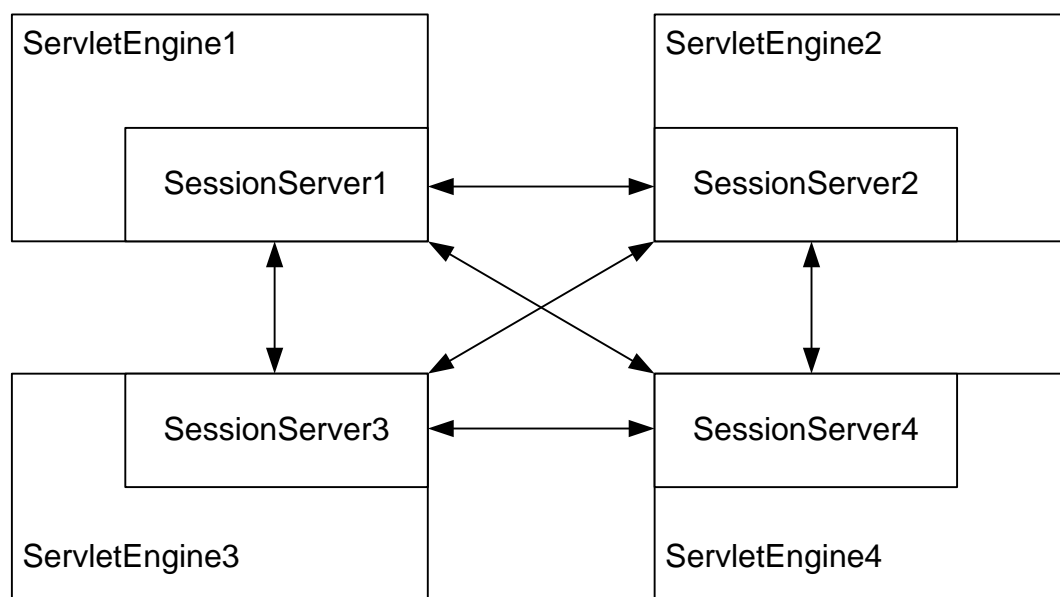
## 2.3. サーバーの構造

分散セッション・サーバーとは、セッション・オブジェクトをサービスするサーバーがそれぞれのサーブレット・エンジン(Webエンジン)、またはEJBエンジンに分散されている分散型の構造です。

分散セッション・サーバー方式は、クラスタリングに参加するすべてのエンジン内に独立した分散セッション・サーバーが存在しており、これらの分散セッション・サーバーがP2P(Peer-to-Peer)方式で別のエンジンの分散セッション・サーバーと通信して持続的なセッション・サービスを提供します。

以下は、分散セッション・サーバー方式を利用し、4つのWebエンジンをセッション・クラスタリングする構造です。

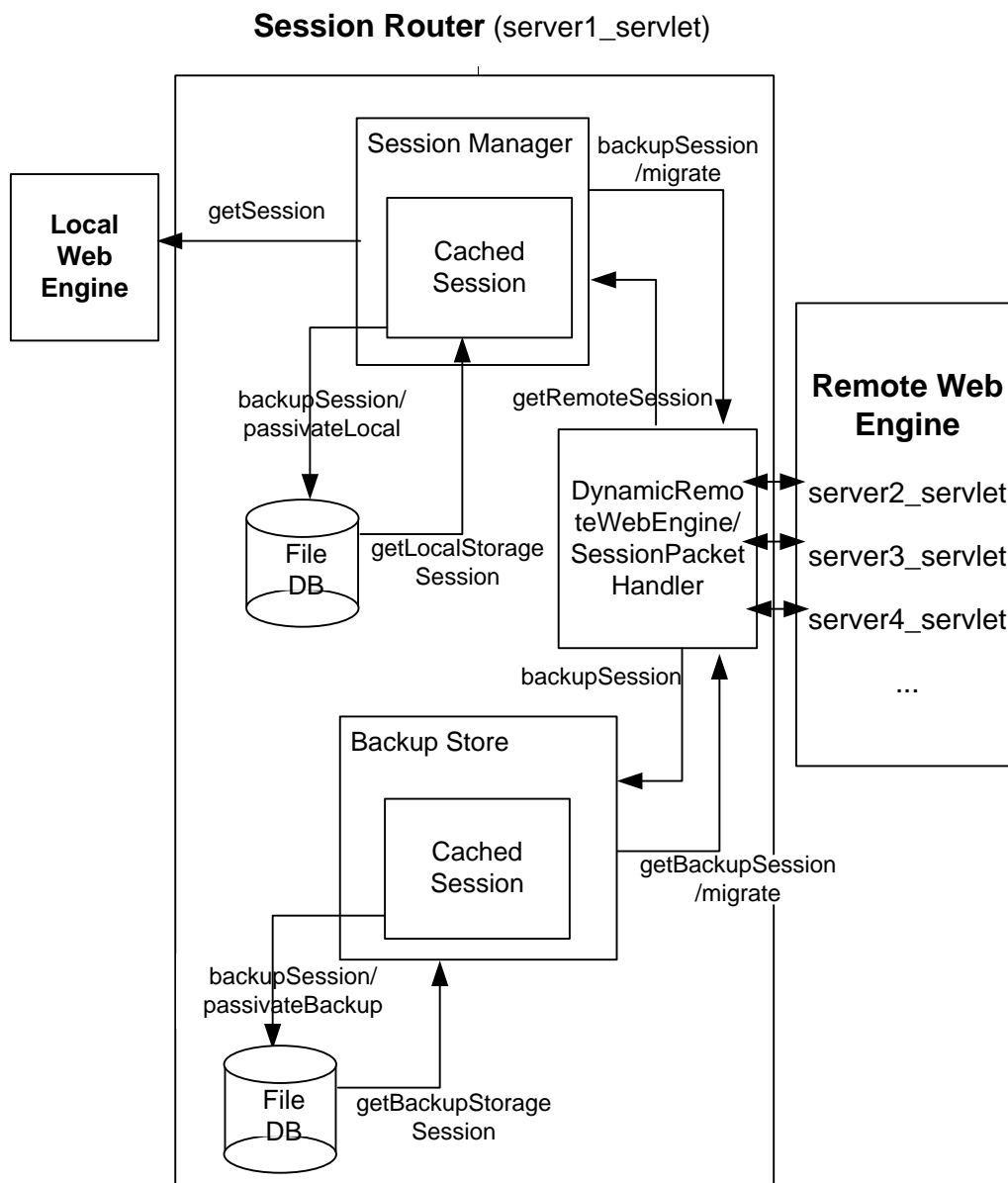
**[図 2.1] 分散セッション・サーバーを利用したセッション・クラスタリングの構造**



上記の構成図において、矢印は分散セッション・サーバー間のソケット接続を表しています。常にすべてを接続するのではなく、接続への可能性を意味しています。一般的にセッションを維持するために別のエンジンとの通信が必要な場合のみ接続します。障害が発生する場合を除いて、通常は1つのみ接続します。

以下は、Webエンジンのサブ・コンポーネントで動作する分散セッション・サーバーの内部構造です。

**[図 2.2] 分散セッション・サーバーの内部構造**



#### ● Session Manager

- エンジンのセッション管理を統括するモジュールです。ローカル・メモリー、ファイルおよび別のリモート分散セッション・サーバーからセッションを取得、管理します。
- ローカルWebエンジンは、getSessionを使用してセッション・マネージャーからセッション・オブジェクトを取得します。

ローカル・メモリーのCached Session、Local File Storage Session(getLocalStorageSession)、Remote Session(getRemoteSession)の順でセッションを取得します。

- メモリーのCached Sessionは、一定の周期(Passivation Timeout)の間使用しないとpassivateLocalによってファイルに保存され、メモリーから削除されます。
- 修正されたセッションは、リモート・バックアップが存在する場合、RemoteWebEngineを通じてバックアップ・サーバーとして指定された別のWebエンジンの分散セッション・サーバーにin-memoryバックアップされます。(backupSession)

修正されたセッションの適用は「**Backup Level**」設定によって異なります。「**Backup Level**」が「all」の場合は常に、「get」の場合はgetterとsetterの呼び出し時、「set」の場合はsetter呼び出し時のみ(setAttribute、removeAttribute、setMaxInactiveInterval)のセッション・オペレーションが発生したとき、修正されたセッションとして判断されます。

「**Backup Level**」の設定値についての詳細内容は、「[2.10.3. 分散型セッション・サーバーの設定](#)」を参照してください。

- 別のリモートWebエンジンからmigrate要求が受信される場合、ローカル・メモリーのCached Session、Local File storageの順でセッションを返します。そのとき、ローカルに存在するセッションを削除し、ownershipを別のリモート分散セッション・サーバーに渡します。

#### ● Backup Store

- 自身の分散セッション・サーバーをバックアップとして選択した他エンジンの分散セッション・サーバーが周期的に転送するバックアップ・セッション・オブジェクトを管理します。このバックアップ・セッション・オブジェクトはプロトタイプを持っているエンジンに障害が発生した場合に備え、セッションを提供します。
- 自身をバックアップとして指定したリモートWebエンジンが転送するバックアップ・セッションを受け、保存および管理します。(backupSession、getBackupSession).
- メモリーのCached Sessionを一定の周期(Passivation Timeout)の間使用しないと、passivateBackupによってファイルに保存され、メモリーから削除されます。
- ローカルで修正されたセッション・オブジェクトは、条件が満たされると指定されたリモート分散セッション・サーバーにバックアップされます。このようなバックアップは、要求が終了すると直ちに実行され、障害状況におけるフェイルオーバーを可能にします。
- 別のリモートWebエンジンからバックアップ・ストアにmigrate要求が受信されると、ローカル・メモリーのCached Session、Local File Storageの順でセッションを返します。

#### ● DynamicRemoteWebEngine

- リモートWebエンジンに存在するセッションに対するオペレーションをコントロールするモジュールです。

該当するエンジンは動的な環境を考慮して運用されます。すなわち、動作中に運用サーバーが追加または削除される場合、状況に適したバックアップが選択、運用されます。この環境設定中心からの脱却は大きな特徴の1つです。リモート・エンジンに対する接続は、pingを介した持続的な確認ではなく、SCF方式を利用して障害状況における不要なエラー状況を除去します。

たとえば、リモートWebエンジンからのgetSession()、removeSession()などがあります。

参考

環境設定中心の既存の構造では、設定によって固定されたバックアップ・エンジンのみを使用しましたが、ダイナミックに変化するリモート・エンジンに対して動的な変更ができるように再設計されました。

- SessionPacketHandler
  - リモートWebエンジンからセッションのオペレーション要求が受信されると、最も先に要求を処理、割り当てするモジュールです。

2.4. 動作方式

分散セッション・サーバー方式は、セッション・ルーティング機能がデフォルトで動作されます。セッションIDに特定のサーバー名を付けて発行しており、サーバーが生成された位置が把握できるので、同情報を利用してセッションを維持します。そのため、セッション・ルーティング機能を強制的に使用しないと、性能の効率が下がります。

セッション・ルーティングをサポートするWebサーバーを前方に配置すると、正常な場合、セッション・オブジェクトが存在するWebエンジンに要求がルーティングされます。したがって、このような場合の動作方式は、セッション・ルーティングによるセッション・クラスタリングと同様です。詳細内容については、「[1.3.2. クラスター環境での動作](#)」を参照してください。

セッション・ルーティング機能がデフォルトで動作される分散セッション・サーバー方式ではセッション・キーを使用します。

以下は、Webエンジンで使用されるセッション・キーの例です。

```
<SessionID>.<primary-engine-name>
例) XXX.domain1/server1
```

項目	説明
XXX	<SessionID>を象徴的に表したもので、実際の<SessionID>はこれより長いランダム・ストリング・タイプです
domain1/server1	ルーティング情報です。domain1というサーバーのサブレット・エンジンを意味します

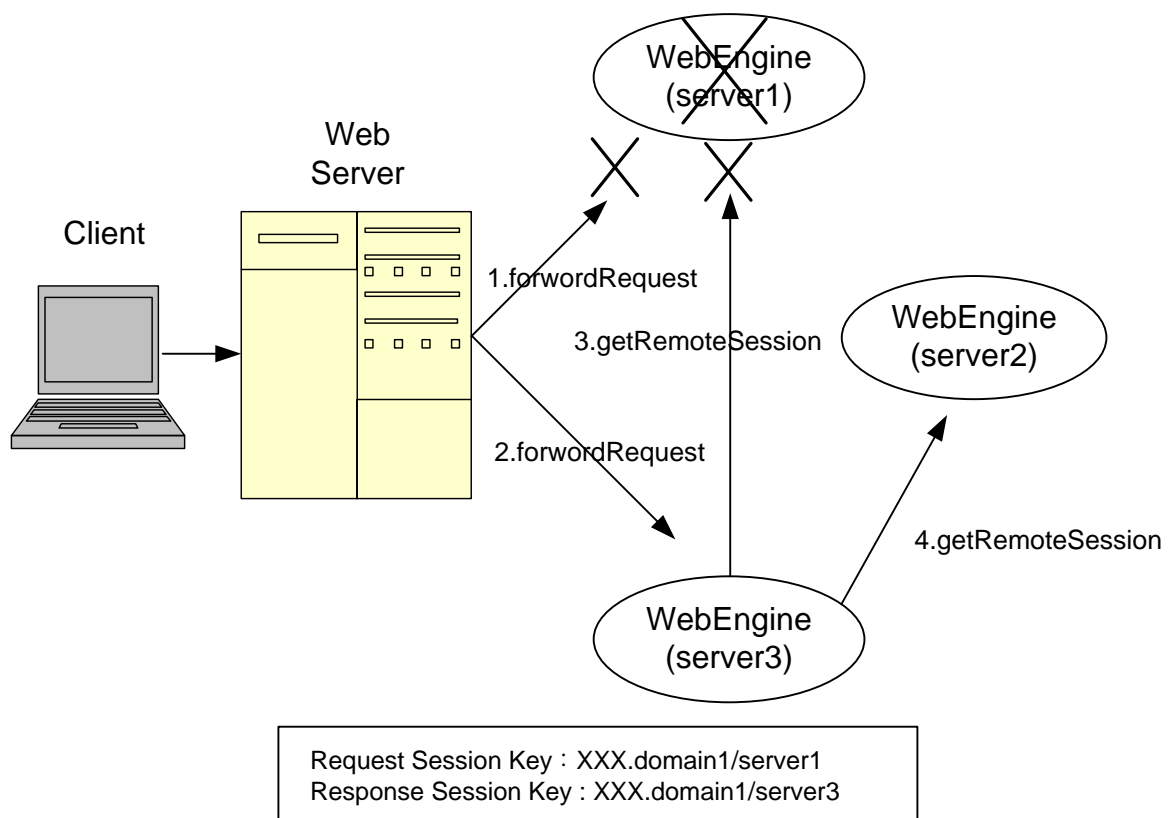
各エンジンには、セッション・ルーティングに使用するセッション・ルーティングIDが1つずつ与えられます。同IDは、WebエンジンがWebサーバーに接続する際にWebエンジンを区分する区切り子として使われます。セッション・ルーティングIDは、設定によって自動生成されます。

[図 2.3]の3つのWebエンジンに、以下のようなセッション・ルーティングIDおよびバックアップ・サーバーのセッション・ルーティングIDが割り当てられていると仮定します。

- WebEngine(server1)
  - セッション・ルーティングID : domain1/server1
  - バックアップ・サーバーのセッション・ルーティングID : domain1/server2
  
- WebEngine(server2)
  - セッション・ルーティングID : domain1/server2
  - バックアップ・サーバーのセッション・ルーティングID : domain1/server3
  
- WebEngine(server3)
  - セッション・ルーティングID : domain1/server3
  - バックアップ・サーバーのセッション・ルーティングID : domain1/server1

セッション・ルーティングIDが割り当てられた分散セッション・サーバーによるセッション・トラッキングのフェイルオーバー・プロセスは以下のとおりです。

[図 2.3] 分散セッション・サーバーによるファイルオーバー・プロセス



以下は、[図 2.3]の動作についての説明です。

1. Webサーバーはセッション・ルーティングIDを分析してWebEngine(server1)に要求を転送しようとしませんが、WebEngine(server1)はフェイル状態です。

2. Webサーバーは別のWebEngineの中から任意で1つを選択して要求を送ります。本例では、WebEngine(server3)を選択しています。

Webサーバーから要求を受けたWebEngine(server3)は、セッション・ルーティングIDを分析します。分析の結果、この要求を処理するセッション・オブジェクトはWebEngine(server1)に存在しており、当該セッションのバックアップはWebEngine(server2)に存在していることが確認できました。

3. 分散セッション・サーバーのWebEngine(server3)は、要求の分析結果に従ってWebEngine(server1)に接続し、セッション・オブジェクトを取得しようとしています(Pimary Migration)。しかし、WebEngine(server1)に障害が発生したため、同要求の処理は失敗します。

4. 要求の処理に失敗したため、WebEngine(server3)はWebEngine(server1)のバックアップ・サーバーであるWebEngine(server2)に再トライします(Backup Migration)。WebEngine(server2)は要求に対し、バックアップしておいたセッション・オブジェクトをWebEngine(server3)に渡します。

5. セッション・オブジェクトの取得に成功したWebEngine(server3)は、クライアントの要求を処理して応答メッセージをクライアントに送ります。その際、新しいセッション・キーを作成して送るので、クライアントのセッ

セッション・キーが変更され、以降の要求はWebEngine(server3)に受信されます。新しいセッション・キーを作成するときは、セッション・ルーティングIDのみ自身のものに置き換えます。

Webサーバーがセッション・ルーティングをサポートしても、Webサーバーにセッション・ルーティングIDに該当するWebエンジンへのコネクションが存在しなかったり、Webエンジンに障害が発生して要求が転送できなかったりする場合、Webサーバーはセッション・ルーティングをサポートすることができません。このような場合のWebサーバーは、任意で選択された別のWebエンジンに要求を渡します。

以下は、セッション・ルーティング機能を使用しない場合における、[\[図 2.3\]](#)の分散セッション・サーバーの動作についての説明です。

1. セッション・ルーティング機能を使用しない場合、セッション・ルーティングIDは存在しません。Webサーバーは任意でWebEngineを選択して要求を送ります。本例では、WebEngine(server1)を選択しています。
2. WebサーバーはWebEngine(server1)がフェイル状態であることを検知し、再び任意でWebEngineを選択して要求を送ります。WebEngine(server3)が選択されました。
3. 分散セッション・サーバーのWebEngine(server3)もまた当該セッションのルーティング情報が分からないため、現在接続中の分散セッション・サーバーに順番どおり要求します。同要求はWebEngine(server1)に送られますが、失敗します。
4. 分散セッション・サーバーは同要求を再びWebEngine(server2)に送り、WebEngine(server2)は該当するセッションを渡します。
5. セッションを受けたWebEngine(server3)は、セッション・キーを変更せずにクライアントに渡すので、次の要求は当該セッションを持っているWebEngine(server3)には送られません。

---

#### 参考

説明のためにルーティング情報をdomain1/server1のように表記していますが、運用中には当該情報がエンコーディングされ動作します。

---



## 2.5. 重複ログイン防止機能

本節では、分散セッション・サーバーに追加された新機能である重複ログイン防止機能について説明します。

重複ログイン防止機能は、アプリケーションで管理する重複ログイン管理機能の基本的な機能を分散型セッション・サーバーで提供するサービスです。

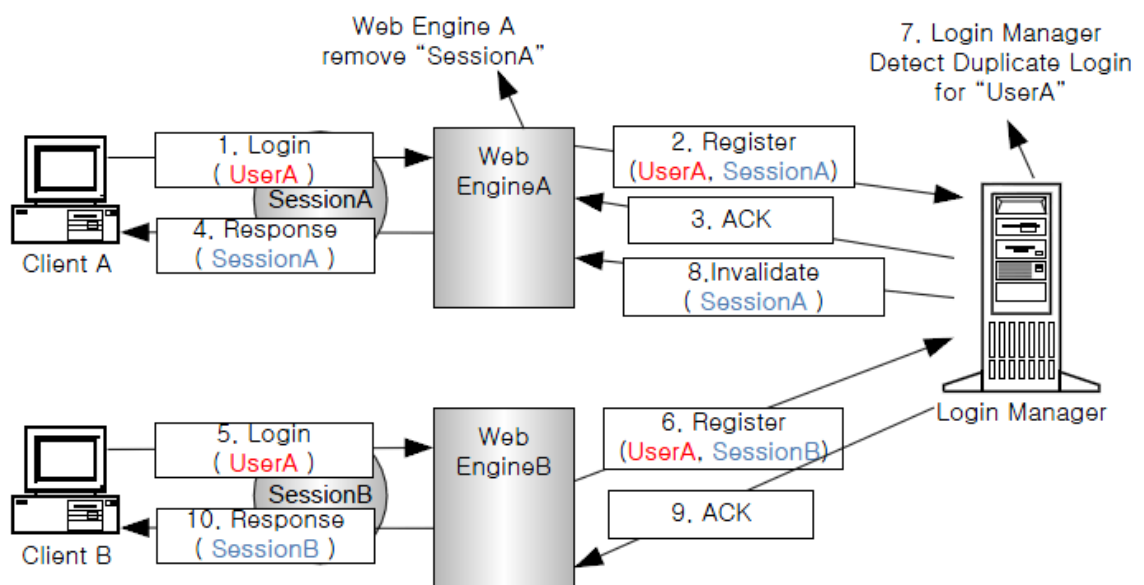
分散型セッション・サーバーの重複ログイン防止機能を使用すると、アプリケーションのIDに対して重複されたログインを許可しません。すなわち、異なるセッションに対して同じIDでログインを実行した場合、既存のログインを実行していたセッションを削除して重複ログインを根本的に防ぎます。

重複ログイン防止機能のために、ログイン・マネージャーは既存の中央セッション・サーバーと同様に構成されます。複数サーバー間のクラスタリングの中からログイン情報を保存するサーバーを指定し、かつ、障害状況を考慮してセカンダリー・サーバーを構成して動作します。

重複ログイン防止機能は、個別のアプリケーション別に設定可能であり、ログイン情報をアプリケーション別に管理します。同設定は、jeus-web-dd.xmlに<use-jeus-login-manager>をtrueに設定します。

重複ログイン防止のプロセスは以下のとおりです。

【図 2.4】 重複ログイン防止機能の動作プロセス



以下は、[【図 2.4】](#)の動作についての説明です。

1. SessionAを使用しているClient AはWeb EngineAにID:UserAでログインをトライします。
2. Web EngineAはLogin Managerにログイン情報を登録します。
3. Login Managerはログインに対するACKをWeb EngineAに渡します。

4. Web EngineAは要求に対するResponseをClient Aに渡します。
5. SessionBを使用している別のClient Bが同じID:UserAでログインをトライします。
6. Web EngineBはLogin Managerにログイン情報を登録します。
7. Login Managerは同じID:UserAに対して重複ログインであることを検知します。
8. Login Managerは既存のログインを実行したSessionAを有するWeb EngineAに対してSessionAを削除するようにコマンドを送信します。Web EngineAは該当するセッションを削除します。
9. Login Managerはログインに対するACKをWeb EngineBに渡します。
- 10 Web EngineBは要求に対するResponseをClient Bに渡します。

---

#### 参考

ログインはアプリケーションではイベントであるため、Jeus Login Managerでログイン・イベントを渡す方法には制約が存在します。Jeus Login Managerでは基本的に「JEUS\_LOGIN\_KEY」でsetAttributeを実行した場合、ログインとして判断します。この値は、jeus-web-dd.xmlのプロパティに「jeus.application.login.key」に変更可能です。同じキーでremoveAttributeを実行した場合はログアウトを実行し、無効になったセッションは自動でログアウトされます。

---

## 2.6. フェイルバック機能

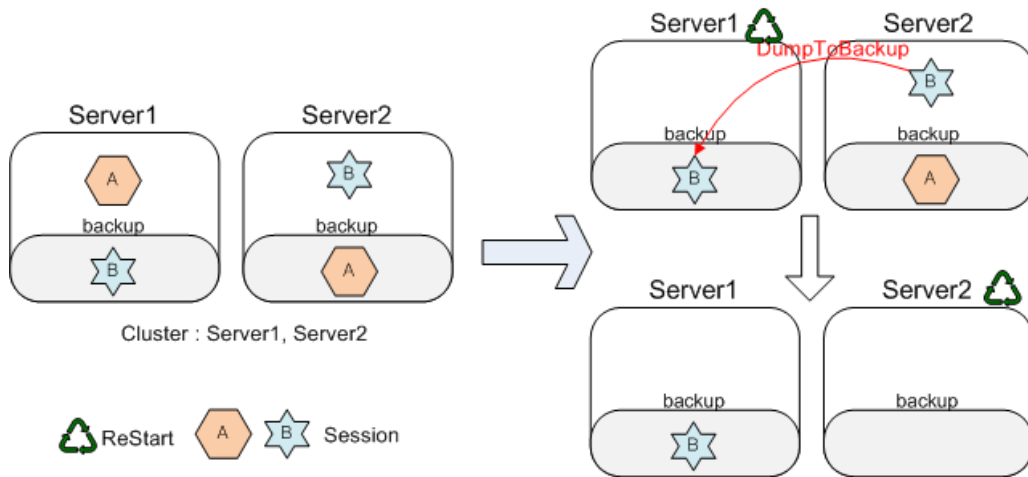
本節では、分散セッション・サーバーのフェイルバック機能について説明します。

フェイルオーバー機能は、障害状況に対応して既存のサービスを続行させるためのセッション・サーバーの基本機能です。すなわち、セッションの作成後セッションをバックアップすることで、特定のサーバーに障害が発生しても、既存のセッションでサービスを続行できる機能です。

新しく提供されるフェイルバック機能は、連続したサーバーの再開または障害サーバーの復旧時に提供されるサービスです。以下では、連続してサーバーを再起動するシナリオでフェイルバック機能について説明します。

下の図は、フェイルバックをサポートしない場合の進行状況を表したものです。

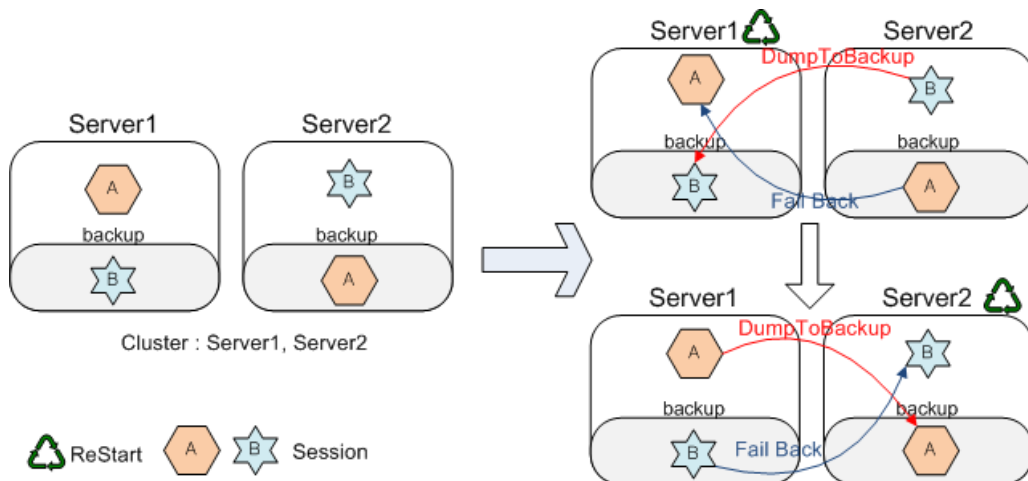
【図 2.5】フェイルバックをサポートしない場合



Server1が再起動すると、Server1が保持していたセッション「A」はServer2のバックアップにのみ存在します。続いてServer2が再起動すると、Server2のバックアップは削除されるため、セッション「A」はシステム全体から削除されてしまいます。以降セッション「A」の要求があると、セッションの維持に失敗し、新しいセッションが作成されます。既存の分散型セッション・サーバーでは、バックアップを取ったセッションの処理を行わなかったためです。

下の図は、フェイルバックをサポートする場合の進行状況を表したものです。

【図 2.6】フェイルバックをサポートする場合



Server1が再起動した場合、DumpToBackupによる動作は同じです(セッション「B」の維持)。次にフェイルバック機能により、Server2が持つセッション「A」が再びServer1に転送されることを確認できます。

続いてServer2が再起動したときも、最初の環境と同様にセッションが維持されることを確認できます。

---

#### 参考

フェイルバックをサポートしなくても、サーバーに障害が発生したときに障害が発生したサーバーのセッションに対するアクセスが発生したら、そのセッションは正常に維持されます(フェイルオーバー機能により)。アクセスされたセッションは障害サーバーが再起動すると、再びバックアップとして転送されるため(dumpToBackup)、後でも正常な動作が可能になります。そのため他ベンダーの一部Webサーバーは既存のすべてのセッションへのアクセスを実行することでセッションを維持する方法を取っています。

---

フェイルバック機能は基本的に設定によってrequest-response方式で実行されます。

バックアップを取った元のサーバーの再起動を検知することで機能が始まります。

1. 特定のサーバーが再起動したら、そのサーバー・セッションのバックアップを取っているかどうかを確認します。
2. 再起動したサーバーがバックアップを取っている元のサーバーである場合、フェイルバックを行うかどうかを聞きます。
3. フェイルバックの設定がある場合、再起動したサーバーはフェイルバック要求を応答として送信します。
4. 元のサーバーから受け取ったバックアップ・セッションを再起動したサーバーに送信します。

---

#### 参考

OOM (Out of memory)による障害が発生した場合には、当機能の使用を推奨しません。セッションの過負荷のため多くのセッションをバックアップした場合、フェイルバックにより継続したOOMを引き起こす可能性があります。フェイルバック機能は、既存のセッションが切れることなくサーバーを順次に再起動したり、ローリング・パッチを適用したりするために提供します。

---

## 2.7. 同時要求セッションの維持機能

本節では、セッションを維持するための分散セッション・サーバーの機能のうち、マイグレーションを実行せずにアップデートする機能について説明します。

セッションは1つの要求内で意味のあるオブジェクトであり、他のJVMや他のスレッドでの同時操作は仕様上では保証されません。

アプリケーションの構成上、同時要求を正常に処理するには、それぞれの要求が異なるセッションを使用するか、セッションを使用しないように構成しなければなりません。しかし、セッションを共有することは大変効率的なので、セッションを共有してアプリケーションを構成する事例が増えつつあります。

このように同時要求がセッションと関連付けられる場合は、以下のような制約が発生します。

1. 新規セッションが同時に作成される可能性があり、そのセッションのうち1つだけがクッキーに保存されるため、メモリーの無駄が発生する恐れがあります。

この制約は、それぞれの要求が異なるJVMに移動する場合や、同じサーバーで同時に処理される場合に共に発生する可能性があります。セッションが存在しない場合、セッションを作成するのが基本的な動作なので、同時に要求が送信されると、それぞれ新しいセッションが発行されます。セッションはユーザーのブラウザのクッキーに保存されますが、クッキーは1つのスコープで管理されるため、最後に発行されたセッションで上書きされます。そのため、別の要求によって作成されたセッションはアクセスできなくなり、タイムアウトによって削除されてしまいます。

2. セッションを同時に別々のJVMで維持する方式をどのように設計するかによってセッションを維持できない場合もあります。

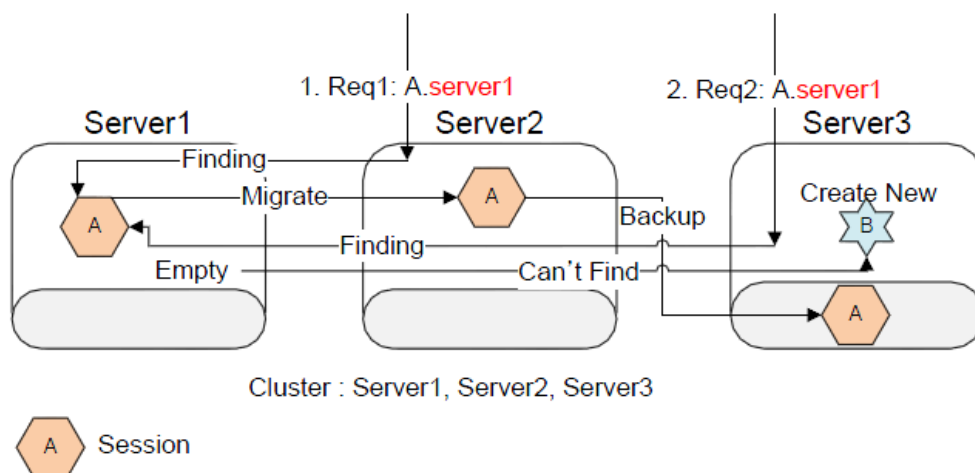
この制約はスティッキー・セッション・ルーティングと関連する制約であり、同時要求が同じサーバーやWebエンジンに送信された場合は該当しません。アプリケーションの構成や環境のため、別々のJVMに要求が同時に送信される場合は、セッションの維持方式をどう開発するかによってセッションが維持されないことがあります。

3. セッションに同時にアップデートを実行する場合、セッションへの反映が無視されることがあります。

第2項と同様に、別々のサーバーやWebエンジンに要求が送信された場合に該当する制約です。セッションのアップデートは1つの要求によって処理されることを仮定しているため、その結果の反映もセッションの作成と同様にタイミングによって反映されないことがあります。

上記の制約事項は、JEUS以外の環境でも同様に発生する可能性があります。JEUSではローカル・セッションを有効に使用するためにマイグレーションを行っています、そのため第2項の制約が選択的に発生することがあります。

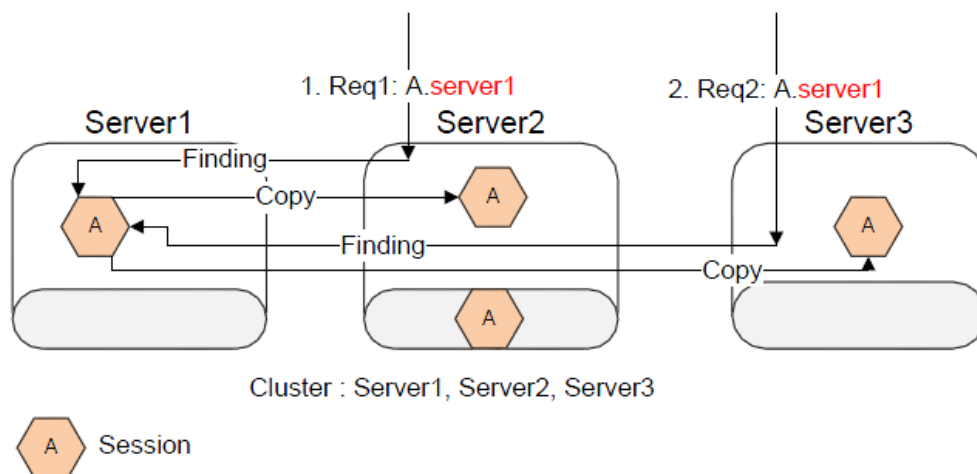
**【図 2.7】セッションの消失が発生する状況**



同時要求が同じサーバーやWebエンジンに送信された場合は、特に問題が発生しません。しかし、アプリケーションの構成によって別々のサーバーに要求が同時に送信された場合は、上の図のような状況が発生します。このようなセッションの消失を防ぐために、ローカル・セッションを使用する効率性のため実行するマイグレーションを実行しない機能が提供されます。

この機能を有効にした場合、以下のように動作します。

**[図 2.8] マイグレーションを実行しない場合の動作**



同時要求が別々のサーバーに送られても、ターゲットになるセッションをマイグレーションせずコピーしてローカルで操作し、その結果を再び元のサーバーにアップデートします。このような動作によりセッションの消失を防止することができ、単純にセッションの値を確認する動作は無理なくサービスできます。しかし、上述した第1項と第3項の制約は同様に存在するため、上記のようなアプリケーションの構成は望ましくはありません。

---

#### 参考

セッションは情報を保存および管理する一時的なオブジェクトであるので、すべてのシステムが参照するデータベースのように使用してはなりません。単にオブジェクトの値を参照する場合にのみ使用し、別々のサーバーに同時要求が送信される構成や設定は控えてください。

---

## 2.8. セッション・クラスター・モード

本節では、セッション・クラスター・モードについて説明します。

セッション・クラスターリングは、セッション情報の共有をサポートする機能です。セッション情報の共有は同じスコープでのみサポートし、スコープが異なる場合は共有を防止することが重要です。

JEUSでは3つのセッション・クラスター・モードをサポートします。モードによってスコープが決定されます。

- デフォルト・セッション・クラスター・モード
- ドメイン・スコープのセッション・クラスター・モード
- 特別定義スコープのセッション・クラスター・モード

## 2.8.1. デフォルト・セッション・クラスター・モード

JEUSがデフォルトで提供するセッション・クラスター・モードです。

サーバーのクラスタリング設定を行うと自動でサポートされます。クラスタリングに参加するサーバー間でセッション・データを共有して使用できます。サードパーティのデフォルト・セッション・スコープの仕様を遵守するため、複数のアプリケーションが存在する場合、相互に独立したセッション管理が可能であり、またセッション・クラスタリングの基本機能である障害復旧やロードバランシングをサポートします。仕様にに基づき基本的なセッション・スコープはアプリケーションになりますが、Webエンジンのセッション・マネージャーが提供するShared設定を利用すると、クラスタリングを構成するサーバー・スコープのセッション・クラスタリングが可能です。

## 2.8.2. ドメイン・スコープのセッション・クラスター・モード

ドメインのサーバー全体にデプロイされるすべてのアプリケーションを同じスコープとして見なすセッション・クラスターをサポートします。

デフォルト・セッション・クラスター・モードはクラスターに依存するため、クラスターの構造に基づく制約があります。その一つが、デプロイ対象をサーバー・クラスターに強制することです。JEUS 6ではコンテナごとに別のアプリケーションをデプロイし、すべてのコンテナのセッションを共有して使用することができました。

以下は、その例です。

- 2つのノードが存在し、それぞれ2つのコンテナを作成します。
- 各コンテナには別のアプリケーションをデプロイします。合計4つのアプリケーションがそれぞれのコンテナに存在します。
- <shared>オプションで同じノード内のセッションを共有し、中央集中型セッション・サーバーを使用して他のノードのセッションを共有します。
- 結果として4つの異なるコンテナの全セッションが共有されます。

デフォルト・セッション・クラスター・モードでは、クラスタリングを構成すると、デプロイの対象がクラスターに強制されるため、すべてのサーバーに同じアプリケーションがデプロイされます。安定したサーバー環境ではこれはリソースの無駄として見られることもあります。しかし、このようなデプロイ方式は、アプリケーションが物理的に特定のサーバーにだけデプロイされ、障害が発生したときにサービスを正常に行えないことを防ぎます。物理的にサーバーごとに別のアプリケーションをデプロイするためには、クラスターに参加してはなりません。

そのため、サーバー・クラスターに参加せずにセッション・クラスタリングを使用したい場合にこのモードを選択します。

ドメイン・スコープ(DOMAIN\_WIDE)モードはドメイン構造の自体仕様とは食い違う、ユーザー利便性を考えた設定であり、最も広範囲かつ便利に使用できますが、次のような制約事項に留意する必要があります。

- EJBとは関係がなく、Webアプリケーションにのみ影響を与えます。

- クラスターリングの設定と同時に設定する場合、クラスターの設定は無視されることがあります。
- このモードを設定すると、ドメイン内(domain.xml)の全サーバーに適用されます。一部のサーバーだけを対象に設定することはできません。
- <shared>オプションがtrueに設定され、すべてのサーバー内の全アプリケーションのセッションが強制的に共有されます。

---

#### 参考

DOMAIN\_WIDEモードを使用する場合、サーバー・クラスターを構成していても、ドメイン・スコープのセッション・クラスター・モードで動作します。

---

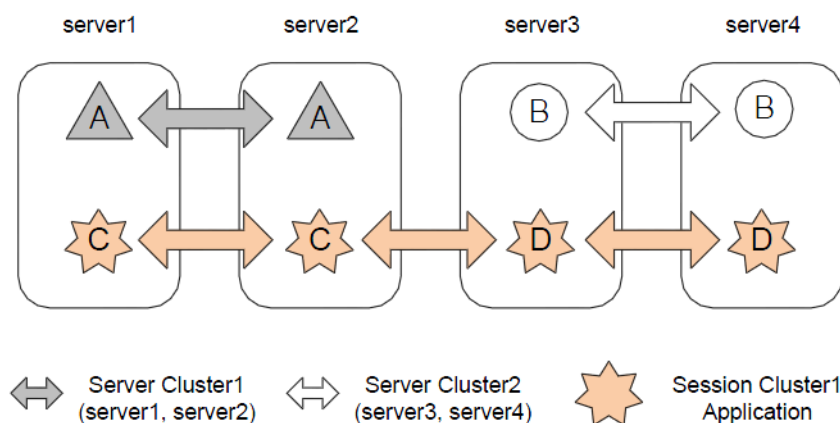
### 2.8.3. 特別定義スコープのセッション・クラスター・モード

特別定義スコープのセッション・クラスター・モードは、設定に基づくセッション・クラスターリングではなく、クラスターの構成と関係なく特定のアプリケーションをグループ化してセッションを共有し、複数のグループが存在する場合に使用するセッション・クラスターリング方式です。ほとんどの場合、共有するグループは1つであるため、デフォルト・セッション・クラスター・モードやドメイン・スコープのセッション・クラスター・モードを使用します。

グループが複数ある場合は、アプリケーションで定義したスコープを選択してデプロイされ、その設定によって維持・管理されます。サポートされるセッション・クラスターリングはスコープだけが異なり、内部の動作は分散セッション・サーバーと同じです。特別定義スコープ機能により、クラスター間のセッション共有や、1つのサーバー内に複数のスコープを構成することが可能です。

下の図は、特別定義スコープのセッション・クラスター・モードのクラスターの構成例です。

**[図 2.9] 特別定義スコープのセッション・クラスターの構成例**



- server1、server2はサーバー・クラスターを構成し、そのクラスターの名前はserver cluster1です。
- server3、serve4はサーバー・クラスターを構成し、そのクラスターの名前はserver cluster2です。



- JEUSのサーバー・クラスターにアプリケーションをデプロイする場合、クラスター内に同じアプリケーションが存在する必要があります。
  - ApplicationA, C : server1, server2
  - ApplicationB, D : server3, server4
- 分散型セッション・サーバーのクラスタリングにより、ApplicationA、ApplicationBはそれぞれのクラスタリングの範囲内でセッションが維持・管理されます。
- 分散型セッション・サーバーの特別定義スコープのセッション・クラスタリングにより、ApplicationC、ApplicationDはサーバー・クラスターを超えたセッションの共有が可能です。すなわち、server1、server2、server3、server4ですべてセッションが維持・管理されます。

上述した特別定義スコープのセッション・クラスターを使用すると、従来のクラスターに限定されたセッションの維持が可能だけでなく、様々なアプリケーションの構成が可能になります。

---

#### 参考

異なるアプリケーション間でセッションを共有するには、基本的にアプリケーション間の共通URLが必要です。これはJEUSの制約事項ではなく、ブラウザとクッキーの制約事項です。そのためセッション・クラスタリングを使用するアプリケーションは共通URLを持ち、このURLをセッション・クッキー・パスに設定する必要があります。

---

## 2.9. セッション・クラスターの基本設定

本節では、セッション・クラスタリングの基本設定について説明します。

セッション・クラスタリングは、セッション情報の共有をサポートする機能です。セッション情報の共有のためにWebエンジンでは分散型セッション・サーバーが運用され、このセッション・サーバーが前節で説明した様々な機能の動作を行います。セッション・クラスタリングの設定には、この分散型セッション・サーバーの運用に関する設定以外に、ドメイン全体に適用される基本設定があります。以下では、この基本設定について説明します。

WebAdminを利用してセッション・クラスタリングの基本設定を行う方法は以下のとおりです。

WebAdminの左側のメニューで**[Session]**を選択すると、セッション・クラスタリングの設定ページに移動します。

[図 2.10] セッション・クラスタリングの設定 - 基本設定

Session

HISTORY

JEUSのセッションクラスタリングの設定です。

ヘルプ

動的設定

必須項目

このページの設定を変更したい場合は、左のメニューから[Lock & Edit]ボタンをクリックしてください。

TIP

Session Cluster

JEUSのセッションクラスタリングに関する設定です。サーブレットで使用するHTTPセッションクラスタリングのためのサービスです。セッションクラスタリングをサポートし、セッションクラスタリングに参加する他のサーバのセッションサーバと通信して、障害が発生した場合にもセッションが維持されるようにします。

Cluster Mode	<div>DEFAULT</div> <p><b>[デフォルト: DEFAULT]</b> JEUSが提供する2つのセッションクラスタモードからいずれかを設定します。 - DEFAULT: 基本的に設定するモードであり、サーバクラスタに依存したセッションクラスタを提供します。サーバクラスタを構成する場合、該当するサーバ間のセッションクラスタをサポートします。構成しない場合は、セッションクラスタが使用されません。 - DOMAIN_WIDE: 全ドメインレベルの1つのスコープでセッションを共有するセッションクラスタを使用する場合に設定します。ドメイン内のすべてのサーバとアプリケーションのセッションが共有されます。特定のサーバにのみデプロイされるアプリケーション間でセッションを共有するときに使用されます。簡単かつ直感的に構成することが可能ですが、デプロイされたサーバがダウンしたとき、柔軟に対処できない場合があります。</p>
Session Manager Provider	<div>JEUS</div> <p><b>[デフォルト: RUNTIME]</b> セッションマネージャのプロバイダを設定します。予約語以外のパッケージ名全体を使用してください。(例: jeus.servlet.sessionmanager.provider.JeusDistributedWebSessionManagerProvider) 予約語は2つを提供します。 - JEUS: JEUSが基本的に提供するセッションマネージャを使用します。 - RUNTIME: 他のプロバイダが存在する場合、そのプロバイダを優先して使用します。存在しない場合は、JEUSのセッションマネージャを使用します。</p>
Exclude Das In Cluster	<div><input type="checkbox"/></div> <p><b>[デフォルト: false]</b> クラスタからDASを除外します。DASをマネージャとして使用するため、セッションクラスタリングの役割を除去します。このオプションはクラスタモードとは関係なく、すべてに適用されます。</p>
properties	<div>key=value</div> <p>ドメインに設定するセッション関連のプロパティ設定です。すべてのスコープのセッションクラスタに同じく適用されます。同じプロパティ設定が下位の特定のセッションクラスタに存在する場合はその設定が使用され、この設定は無視されます。</p>
Sticky Encoding Rule	<div>RAW</div> <p><b>[デフォルト: BASE64]</b> スティッキーエンコーディングのルールをサポートします。現在は、BASE64とRAWをサポートしています。 - BASE64: Base64を使用します。 - RAW: エンコーディングなしで、ドメイン名またはサーバ名を使用します。</p>

セッション画面は**Session Cluster**領域と**Specific Scope Cluster**領域で構成されます。**Session Cluster**領域は、ドメイン全体に適用される設定とCommon Cluster Configの設定に区分されます。

## ● Session Cluster

項目	説明
Cluster Mode	<ul style="list-style-type: none"> <li>DEFAULT: サーバー・クラスターを構成する場合はセッション・クラスターをサポートし、サーバー・クラスターを構成しない場合はセッション・クラスターをサポートしません。基本的にアプリケーション単位のセッション・クラスターがサポートされます。特別定義スコープ(SPECIFIC_SCOPE)セッション・クラスター・モードを使用する場合もDEFAULTモードを設定します(デフォルト値)</li> <li>DOMAIN_WIDE: ドメインの全サーバーのすべてのアプリケーションのセッションを共有するセッション・クラスターをサポートします</li> </ul>

項目	説明
	それぞれのセッション・クラスター・モードはスコープが異なるだけで、同じ形式の分散型セッション・サーバーの設定を使用します。各モードのクラスタリング参加方式および分散セッション・サーバーの設定方法については「 <a href="#">2.10. セッション・クラスター・モード別の設定</a> 」で説明します
Session Manager Provider	<p>セッション・マネージャーとして使用する実装または予約語を設定します。予約語以外にはパッケージの全体の名前を設定します。</p> <p>予約語にはRUNTIMEとJEUSがあります</p> <ul style="list-style-type: none"> <li>– RUNTIME : JEUSが提供するプロバイダー以外に他の実装が存在すれば、その実装を使用するように設定します(デフォルト値)</li> <li>– JEUS : JEUSが提供する他の実装が存在しても、JEUSの実装を使用します</li> </ul>
Exclude Das in Cluster	<p>クラスターにDASを強制に参加させたくない場合に設定します。ドメイン・スコープ・モードを設定した場合に意味があります</p> <ul style="list-style-type: none"> <li>– true : クラスターにDASが含まれていても、クラスタリングに参加せずにスタンドアローンで動作します</li> <li>– false : DASは他のサーバーと同様に扱われ、クラスタリングへの参加の有無に合わせて動作します(デフォルト値)</li> </ul>
Sticky Encoding Rule	<p>スティッキー・ルーティングの対象となる情報のエンコーディング・ルールをサポートします。現在は次の2つをサポートします</p> <ul style="list-style-type: none"> <li>– BASE64 : スティッキーされる情報をBASE64のルールに従ってエンコーディングして転送します(デフォルト値)。セッションIDによりエンジン情報やドメイン情報が露出しないように、意味のない情報に見せかけるためにエンコーディングします</li> <li>– RAW : スティッキーされる情報をエンコーディングせずにそのまま転送します。エンコーディングされた情報は内部でもどのエンジンからの要求かが判断し難いため、デバッグする場合や、エンジン名の露出がセキュリティに影響を与えない場合にこの値を設定します</li> </ul>
Properties	ドメインのすべてのクラスターおよびセッション・サーバーに適用されるプロパティの設定です。プロパティ項目はキーと値で構成されます

## 2.10. セッション・クラスター・モード別の設定

本節では、セッション・クラスター・モード別のクラスタリング参加方式や分散型セッション・サーバーの設定位置、分散型セッション・サーバーの共通設定について説明します。

### 2.10.1. セッション・クラスター・モード別のクラスタリング参加方式

セッション・クラスター・モード別にセッション・クラスターに参加する方式を説明します。

#### デフォルト・セッション・クラスター・モード

デフォルト・セッション・クラスター・モードでセッション・クラスタリングに参加する方法はとても簡単で、サーバー・クラスタリングを設定するだけで参加できます。

1. WebAdminの左側のメニューから[Clusters]を選択すると、**クラスター・リストの照会画面**が表示されます。
2. [LOCK & EDIT]ボタンをクリックして、設定変更モードに切り替えます。
3. クラスタリングを追加するには[Add]ボタンをクリックし、既存のクラスタリングを変更するには[Clusters]画面の照会されたリストから目的のクラスターを選択します。

**[図 2.11] クラスタリング - クラスター・リストの照会**



4. 「Servers」項目にドメイン内のサーバーが表示されます。以下のようにクラスターを構成するサーバーを選択した後、[確認]ボタンをクリックします。

[図 2.12] クラスタリング - クラスターを構成するサーバーの選択および確認

5. 設定を完了した後、変更内容を動的に反映するために[**Activate Changes**]ボタンをクリックします。
6. 設定内容が反映されると、以下のように結果メッセージが出力されます。

[図 2.13] クラスタリング - 結果の確認

```
clusters.cluster.{? name == 'distribute' }.servers.serverName : ACTIVATED
clusters.cluster.{? name == 'distribute' }.servers.serverName.{? serverName == 'server3' } : ACTIVATED
previous value : null, edited value : server3, result value : server3
clusters.cluster.{? name == 'distribute' }.servers.serverName.{? serverName == 'server4' } : ACTIVATED
previous value : null, edited value : server4, result value : server4
```

## 参考

分散セッション・サーバーは内部的に自動で実行され、設定内容が動的に反映されます。しかし、アプリケーション・スコープで実行されるセッション・サーバーは、デプロイされるアプリケーションによって再起動が必要な場合があります。

## ドメイン・スコープのセッション・クラスター・モード

以下では、ドメイン・スコープ(DOMAIN\_WIDE)のセッション・クラスター・モードの設定方法を説明します。

1. WebAdminの左側のメニューから[Session]を選択すると、**Session画面**が表示されます。
2. [LOCK & EDIT]ボタンをクリックして、設定変更モードに切り替えます。
3. **Session画面**で「Cluster Mode」項目を「DOMAIN\_WIDE」に選択します。

【図 2.14】セッション・クラスター・モード：DOMAIN\_WIDE



4. 設定が完了した後、変更内容を反映するために[Activate Changes]ボタンをクリックします。
5. 変更内容が反映されると、結果メッセージが出力されます。動的設定項目ではないので、変更内容をサーバーに反映するには、サーバーを再起動する必要があります。

## 特別定義スコープのセッション・クラスター・モード

特別定義スコープ(SPECIFIC SCOPE)のセッション・クラスタリングは、DEFAULTモードのサブモードとして見ることができます。すなわち、DEFAULTモードを基本として詳細設定を行います。

WebAdminで特別定義スコープは以下のように設定します。

1. WebAdminの左側のメニューから[Session]を選択すると、**Session画面**が表示されます。

[図 2.15] セッションの選択 - Session画面



2. セッション・クラスタリングを追加するには、下部にある[Add]ボタンをクリックすると、Clustersの**Session Cluster Config**設定画面と同じ画面が表示されます。この画面で同様に個別のセッション・クラスタリングを設定できます。

[図 2.16] セッション・クラスタリングの設定 - セッション・クラスターの追加



3. 個別のセッション・クラスタリングを適用するために、再度確認した後[Activate Changes]ボタンをクリックします。
4. 当該セッション・クラスターにアプリケーションをデプロイするには、jeus-web-dd.xmlの<target-session-cluster>にセッション・クラスタリングを適用するセッション・クラスターを設定します。

#### 参考

セッション・クラスタリングはアプリケーション・レベルで決定するので、必ずjeus-web-dd.xmlにセッション・クラスターを1つ選択して設定してください。

## 2.10.2. 分散型セッション・サーバーの設定位置

分散型セッション・サーバーはWebAdminの次の3つの画面で設定することができます。

- [Common Cluster Config](#)
- [Cluster Session Cluster Config](#)
- [Specific Scope Session Cluster Config](#)

## Common Cluster Config

WebAdminの左側のメニューで[Session]を選択すると、**Session画面**が表示されます。

[図 2.17] セッションの選択 - Session画面



画面の基本設定([図 2.10])の下に共通設定の領域があります。残りの項目についての説明は「[2.10.3. 分散型セッション・サーバーの設定](#)」を参照してください。

[図 2.18] 分散型セッション・サーバーの設定 - 共通設定領域

<b>Common Cluster Config</b>	
セッションクラスタリングの基本的な設定です。各クラスタリングに詳細設定をしない場合は、この設定値が適用されます。- DEFAULT: サーバクラスターにセッションクラスタを設定する場合、この設定は無視され、各サーバクラスターの設定に従います。- DOMAIN_WIDE: この設定に従ってセッションクラスタリングが動作します。- SPECIFIC_SCOPE: Specific Scope Clustersを設定しないと、この設定によりセッションクラスタリングが動作します。個別に設定した場合は無視されます。	
Reserved Thread Num	0 [デフォルト: 0] 分散型セッションサーバに送られた要求を処理するためのスレッドプールの付加設定を行います。基本的にはシステムスレッドプール(Threadpool.System)を使用しますが、このサービスを行うためのスレッドを予め割り当てる必要がある場合は設定を行います。この設定値と他のサービスの値の合計がシステムスレッドプールの最大値を超えてはいけません。
Connect Timeout	5000 ms [デフォルト: 5000] Webエンジンに存在するセッションサーバ間のソケットコネクションを作成時に適用されるタイムアウト値を設定します。単位はミリ秒です。
Read Timeout	20000 ms [デフォルト: 20000] Webエンジンに存在するセッションサーバ間の通信に適用される読み取りタイムアウト値を設定します。データを送ってから、設定した時間の間応答を待ちます。単位はミリ秒です。

**Common Cluster Config**の設定は分散型セッション・サーバーの基本設定で、セッション・クラスターの基本設定(「[2.9. セッション・クラスターの基本設定](#)」)のCluster Mode設定によって適用の可否が決定されます。

区分	説明
DEFAULT	サーバー・クラスターの分散型セッション・サーバーの設定が存在しない場合、この共通設定が使用されます。サーバー・クラスターが構成されていなかったり、サーバー・クラスターに設定が存在すれば、この共通設定は無視されます
DOMAIN_WIDE	この共通設定がドメイン全体に適用される分散型セッション・サーバーの設定として使用されます
SPECIFIC_SCOPE	SPECIFIC_SCOPEの設定が存在しない場合、この共通設定が使用されます。SPECIFIC_SCOPEモードのSpecific scope cluster内の設定を別途に行った場合、この設定は無視されます



## Cluster Session Cluster Config

WebAdminの左側のメニューで[Clusters]を選択すると、クラスター・リストが照会されます。

[図 2.19] クラスタリング - クラスター・リストの照会



クラスターを選択した後、[Session Cluster Config]タブを選択すると、分散型セッション・サーバーの設定を行います。

[図 2.20] 分散型セッション・サーバーの設定 - サーバー・クラスターの設定位置



この分散型セッション・サーバーの設定は、DEFAULTモードでのみ意味があります。クラスターに分散型セッション・サーバーの設定が存在する場合に当該設定が使用されます。設定がない場合は、共通設定が使用されます。各サーバー・クラスターの分散型セッション・サーバーを別々の設定で運用したい場合は、それぞれを設定すると適用されます。別々に設定しない場合は、1つの共通設定で複数のサーバー・クラスターの運用が可能です。画面についての詳細な説明は「[2.10.3. 分散型セッション・サーバーの設定](#)」を参照してください。

## Specific Scope Session Cluster Config

WebAdminの左側のメニューで[Session]を選択すると、基本設定([図 2.10](#))の一番下のSpecific Scope Clusterを照会できます。

【図 2.21】 Specific Scope Cluster - クラスター・リストの照会

Specific Scope Cluster	
Name	Add
specific1	Delete
specific2	Delete

クラスターの名前を選択すると、対応する分散型セッション・サーバーの設定が照会されます。

【図 2.22】 分散型セッション・サーバーの設定 - 特別定義スコープの設定位置

Specific Scope Cluster

HISTORY

特定のアプリケーションでグループを設定し、そのグループスコープでセッションを共有及び維持するときに設定します。 JEUSのセッションクラスタリングでは、該当のグループを独立したアプリケーションと見なし、セッションを共有します。 このような共有を可能にするには、ブラウザのクッキーが他のグループのスコープとは別に運用される必要があります。 グループ間でクッキーパスは一意の値で設定される必要があり、グループ内のアプリケーションはその値で共通してコンテキストパスを適用しなければならない制約があります。 スコープを設定し、そのスコープ別に異なるクラスタ設定を適用するときに設定します。

ヘルプ

Basic

Session Cluster Config

動的設定

※ 必須項目

確認

再設定

Name	<div>specific1</div> <div>           設定したスコープの一意の名前を設定します。 この名前はドメイン内で一意である必要があり、セッションクラスタを構成する際、一意の識別子(ID)として使用されます。 各アプリケーションで該当の識別子を選択してグループを決定します。         </div>
------	---

確認

再設定

この分散型セッション・サーバーの設定は、SPECIFIC SCOPEモードを使用する場合にのみ意味があります。各スコープごとに分散型セッション・サーバーを別の設定で運用したい場合は、それぞれを設定すると適用されます。設定しない場合は、1つの共通設定で複数のSPECIFIC SCOPEのセッション・クラスターの運用が可能です。画面についての詳細な説明は「[2.10.3. 分散型セッション・サーバーの設定](#)」を参照してください。

### 2.10.3. 分散型セッション・サーバーの設定


以下は、分散型セッション・サーバーの設定画面と項目についての説明です。

[図 2.23] 分散型セッション・サーバーの設定

Reserved Thread Num	<div>0</div> <p>[デフォルト:0] 分散型セッションサーバに送られた要求を処理するためのスレッドプールの付加設定を行います。基本的にはシステムスレッドプール(Threadpool.System)を使用しますが、このサービスを行うためのスレッドを予め割り当てる必要がある場合は設定を行います。この設定値と他のサービスの値の合計がシステムスレッドプールの最大値を超えてはいけません。</p>
Connect Timeout	<div>5000ms</div> <p>[デフォルト:5000] Webエンジンに存在するセッションサーバ間のソケットコネクションを作成時に適用されるタイムアウト値を設定します。単位はミリ秒です。</p>
Read Timeout	<div>20000ms</div> <p>[デフォルト:20000] Webエンジンに存在するセッションサーバ間の通信に適用される読み取りタイムアウト値を設定します。データを送ってから、設定した時間の間応答を待ちます。単位はミリ秒です。</p>
Allow Fail Back	<div><input checked="" type="checkbox"/></div> <p>[デフォルト:true] フェイルバックを許可するかどうかを設定します。障害が発生したサーバが再起動される際、バックアップしておいたセッションを再び読み込むかどうかを決定します。起動されるサーバの設定に応じて動作が決まります。</p>
backup-level	<div>access</div> <p>[デフォルト:access] セッションバックアップの実行方式やバックアップの有無についての基準を設定します。</p>
Backup Unit Size	<div>50</div> <p>[デフォルト:50] 内部でバックアップを実行するセッションの単位数です。通常のアップデートは1つのセッション単位で行われますが、負荷状況下では、一度に複数のセッションが送られます。同時に送られる最大セッション数を設定します。ネットワークパケットとセッションのサイズを考慮する必要があります。</p>
Backup Queue Size	<div>20</div> <p>[デフォルト:20] ネットワークが不安定な場合、バックアップ転送が遅延することがあります。遅延が発生した場合でも、キューにセッションを格納してサーバレットを実行します。キューがいっぱいになったらワーカが待機することになるので、フローコントロールが行われます。キューにはバックアップユニット単位で格納されます。backup-unit*backup-queue-size分のセッションが格納できます。</p>
Ignore Flow Control	<div><input type="checkbox"/></div> <p>[デフォルト:false] バックアップキューがいっぱいになっても、継続してサービスを実行する場合に設定します。該当のセッションはバックアップされていないので、障害が発生したら情報が失われる可能性があります。障害状況やルーティングエラーより、サービスの実行が優先する場合に設定します。</p>
Prevent Migration	<div><input type="checkbox"/></div> <p>[デフォルト:false] JEUSにおけるセッションの維持方式は、マイグレーションを基本として動作します。障害や負荷によって、セッションが存在しないサーバに要求が渡された場合、当該サーバに再び同様な要求が送られることに備え、セッションに対してルーティング情報のオーナーシップを変更してから応答を返します。ただし、応答が受信される前に複数のサーバに同時に要求を送信してサービスを実行するなどのサーバレット仕様を超える状況では、セッションが失われる可能性があります。このような状況に備えて、マイグレーションを実行しない方式を提供しています。この設定を使用すると、最初にセッションが作成された場所へのアップデートが継続して行われるので、パフォーマンス上のデメリットが生じます。</p>
Failover Delay	<div>600s</div> <p>[デフォルト:600] Webエンジンに障害が発生した場合、そのエンジンを除く残りのエンジンで再びクラスタリング接続を確立するまでのタイムアウト値を設定します。つまり、この設定値は障害状況で障害エンジンの復旧を待機する時間になります。単位は秒です。</p>
Restart Delay	<div>600s</div> <p>[デフォルト:600] Webエンジンを正常に終了した場合、そのエンジンを除く残りのエンジンで再びクラスタリング接続を確立するまでのタイムアウト値を設定します。エンジンを終了する主な理由は再起動のためであり、この設定は再起動の性能を向上させるためのものです。単位は秒です。</p>
properties	<div> <div>key=value</div> <div> <p>セッションクラスタスコープに適用されるプロパティ設定です。ここに設定を追加すると、同じ名前のドメインレベルのプロパティは無視されます。該当するクラスタにのみ設定を行うときに追加してください。</p> </div> </div>

#### ▼ Jeus Login Manager

JEUSログインマネージャのプライマリサーバとセカンダリサーバを設定します。クラスレベルで重複するログインを防止するためのJEUSログインマネージャのログイン情報が保存されるサーバを設定します。

Primary 	<div></div> JEUSログインマネージャのプライマリサーバを設定します。
Secondary	<div></div> JEUSログインマネージャのセカンダリサーバを設定します。

#### ▼ Passivation

一定時間が経過したり特定の条件を満たす場合に、メモリに保持していたセッションをファイルに保存してメモリの効率を高める機能です。この操作は、基本的にアプリケーション別に行われますが、jeus-web-dd.xml、domain.xmlのサーバ設定の<session-config><shared>がtrueの場合はサーバ別に行われます。設定しない場合は、すべてのセッションをメモリに保持します。

File Path	<div></div> セッションを格納するファイルパスを絶対パスで指定します。ファイルバックアップは、jeus-web-dd.xml、domain.xmlのサーバ設定の<session-config><shared>がtrueの場合は\$(SERVER_HOME)/workspace/session/distributed/<server_name>、falseの場合は\$(SERVER_HOME)/workspace/session/distributed/<context_name>と設定されます。
Single Folder File Limit	<div>10000</div> [デフォルト: 10000] 1セッション当たり1つのファイルを使用します。1つのフォルダに格納されるセッションファイルの数を制限します。設定値が低すぎると、不要なフォルダが多く作成される可能性があり、10000個以上を設定した場合は、OSに応じてファイルI/Oの性能が低下することがあります。
Min Hole	<div>100</div> [デフォルト: 100] ファイルDBを一定時間運用すると、ファイルのサイズが大きくなります。この設定はファイルのサイズが必要以上に大きくなることを防ぐためのもので、ファイルI/Oの発生が指定した回数になると、ファイルパッキングを行います。
Packing Rate	<div>0.5</div> [デフォルト: 0.5] ファイルDBを一定時間運用すると、ファイルのサイズが大きくなります。この設定はファイルのサイズが必要以上に大きくなることを防ぐためのもので、現在のセッションオブジェクト数に対するファイルI/O回数の割合が指定の値を超えると、ファイルパッキングを行います。
Ratio	<div>0.7</div> [デフォルト: 0.7] セッションの数とセッションのメモリをトリガとしてパッシベーションを行うとき、パッシベーションの割合を設定します。設定した割合に従って、セッション数やメモリ量を維持するようにパッシベーションを行います。タイムアウトによってパッシベーションを実行するときは適用されません。

#### ▼ Trigger

ファイルパッシベーションを実行するトリガを設定します。timeout、session count、session memoryを提供しています。

Timeout	<div>-1</div> [デフォルト: -1] メモリに存在するセッションオブジェクトを一定時間使用しない場合、ファイルに保存する設定です。-1または0の場合は、ファイルへのパッシベーションを実行しません。0より大きく設定すると、指定された時間以上使用していないメモリのセッションオブジェクトはファイルにパッシベーションされます。単位はミリ秒で、デフォルト値は-1です。
Count Threshold	<div>-1</div> [デフォルト: -1] メモリ上に存在するセッションが一定数を超えた場合、一部のセッションをファイルに保存します。この操作は、一定周期ごとに実行されるモニタリングスレッドによって行われます。セッション数を設定します。-1または0を設定する場合は実行しません。
Memory Threshold	<div>-1</div> [デフォルト: -1] セッションが一定のメモリを占める場合、一部のセッションをファイルに保存します。この操作は、一定周期ごとに実行されるモニタリングスレッドによって行われます。直列化を実行した結果のバイト数を設定します。-1または0を設定する場合は実行しません。

● 基本情報

項目	説明
Reserved Thread Num	<p>分散型セッション・サーバーに送られた要求を処理するためのスレッド・プールの付加設定を行います。</p> <p>基本的にはシステム・スレッド・プールを使用するので、特に設定する必要はありません。同サービスのためのスレッドを事前に割り当てる必要がある場合にのみ設定してください</p>
Connect Timeout	Webエンジンに存在するセッション・サーバー間のソケット・コネクションを生成するときに適用されるタイムアウト値です
Read Timeout	Webエンジンに存在するセッション・サーバー間の通信に適用される読み取りタイムアウト値です。データを送信してから設定した時間の間応答を待ちます
Allow Fail Back	JEUSが提供するフィエルバック機能を使用するかどうかを設定します
Backup Level	<p>使用されたセッションをリモートWebコンテナまたはローカル・ファイル・データベースにバックアップするときに適用する基準を設定します</p> <ul style="list-style-type: none"> <li>– <code>access</code> : セッションの属性別にアップデートを行い、該当するオブジェクトは <code>setAttribute/putValue/removeAttribute/removeValue/getAttribute/getValue</code> 関数で呼び出した特定の属性のみをバックアップします(デフォルト値)</li> <li>– <code>set</code> : セッションの <code>setAttribute/putValue/removeAttribute/removeValue</code> 関数の呼び出しが発生した場合にのみアップデートと見なし、該当するセッション・オブジェクトをバックアップします(デフォルト値)</li> <li>– <code>get</code> : セッションの <code>setAttribute/putValue/removeAttribute/removeValue/getAttribute/getValue</code> 関数の呼び出しが発生した場合にのみアップデートと見なし、該当するセッション・オブジェクトをバックアップします。</li> <li>– <code>all</code> : 使用されたセッションをすべてバックアップします。セッション・オブジェクトが <code>HttpServletRequest.getSession()</code> APIで呼び出される場合にアップデートと見なし、該当するセッション・オブジェクトをバックアップします</li> </ul>
Backup Unit Size	セッションのバックアップを実行する時の単位を設定します。同時に送信するセッションの最大数を設定します
Backup Queue Size	<p>ネットワークの遅延のため、バックアップが正常に送信されなかった場合に待機させるキューのサイズを設定します。</p> <p>分散型セッション・サーバーでバックアップが存在しない場合は、障害状態でセッションの維持に失敗するため、該当するキューがいっぱいになると、要求は自然に遅延してフロー・コントロールされます</p>

項目	説明
Ignore Flow Control	バックアップ・キューがいっぱいになっても、サービスを続行したい場合に設定するオプションです。セッションの消失よりサービスが優先する場合に使用します
Prevent Migration	JEUSの同時要求機能を使用する場合もセッションを維持します。セッションは維持されますが、この機能を使用する場合、最初にセッションが作成されたサーバーに継続してアップデートを実行しなければならないため、性能上の不利益をもたらします
Failover Delay	障害状況でエンジンの復旧を待つ時間を設定します。  Webエンジンに障害が発生した場合、そのエンジンを除く残りのエンジンで再度クラスタリング接続を確立するまでのタイムアウト値です
Restart Delay	Webエンジンが正常に終了したとき、そのエンジンを除く残りのエンジンで再度クラスタリング接続を確立するまでのタイムアウト値です。エンジンを終了する主な理由は再起動であり、再起動の性能を高めるための設定です
Properties	セッション・クラスターに適用されるプロパティの設定です。プロパティ項目はキーと値で構成されます

#### ● Jeus Login Manager

重複ログインの防止機能を実行するJeus Login Managerのプライマリー・サーバーとセカンダリー・サーバーを設定します。クラスター・レベルの重複ログインの防止のため、JEUS Login Managerがログイン情報を保存するサーバーを設定します。

項目	説明
Primary	JEUS Login Managerのプライマリー・サーバーを設定します
Secondary	JEUS Login Managerのセカンダリー・サーバーを設定します

#### ● Passivation

アップデートされたセッション・オブジェクトをファイル・システムに保存してバックアップする方法を設定します。<shared>がtrueに設定された場合はサーバー別に行われ、falseに設定された場合はアプリケーション別に行われます。

以下のような詳細項目を設定できます。

項目	説明
File Path	セッションを保存するファイル・パスを絶対パスで設定します。  ファイルのバックアップ単位は、jeus-web-dd.xml、domain.xmlのサーバー設定内の<shared>設定に従って、次の値が設定されます  – true :

項目	説明
	<p>"\${SERVER_HOME}/.workspace/session/distributed/{server_name}"</p> <p>– false :</p> <p>"\${SERVER_HOME}/.workspace/session/distributed/{context_name}"</p>
Single Folder File Limit	<p>1セッションあたり1つのファイルを使用します。</p> <p>1つのフォルダーに保存されるセッション・ファイルの数を制限します。少なすぎる数を設定すると、不要なフォルダーが多く作成されることがあり、10000以上を設定すると、オペレーティング・システムによってはファイルの入出力性能の低下が発生することがあります</p>
Min Hole	<p>一定の時間ファイルDBを運用するとファイルのサイズが大きくなるので、設定した基準に従ってファイル・パッキングを実行し、ファイルのサイズが必要以上に大きくなることを防ぎます。</p> <p>ファイル・パッキングを実行する基準となるファイルI/Oの回数を設定します</p>
Packing Rate	<p>一定の時間ファイルDBを運用するとファイルのサイズが大きくなるので、設定した基準に従ってファイル・パッキングを実行し、ファイル・サイズが必要以上に大きくなることを防ぎます。</p> <p>ファイル・パッキングを実行する基準となるセッション・オブジェクト数に対するファイルI/O回数の割合を設定します</p>
Ratio	<p>セッションの数やセッションのメモリーを基にパッシベーションを行う場合、パッシベーションを実行するセッションの割合を設定します。</p> <p>設定した割合のセッションの数またはメモリー量を維持するようにパッシベーションを行います。タイムアウトによってパッシベーションを実行するときは適用されません</p>
Trigger	<p>– Timeout : メモリーにあるセッション・オブジェクトを一定時間使用しなかった場合にファイルに保存する設定です。-1または0を指定すると、ファイルにパッシベーションしません。0より大きい値を設定すると、指定した時間以上使用していないメモリー上のセッション・オブジェクトをファイルにパッシベーションします(単位: ミリ秒、デフォルト値: -1)</p> <p>– Count Threshold : セッションが一定数以上メモリーに存在する場合、セッションの一部をファイルに保存します。この作業は一定周期で実行されるモニタリング・スレッドによって実行されます。セッション数を設定します。-1または0を指定すると、実行しません</p>

項目	説明
	<ul style="list-style-type: none"> <li>Memory Threshold: セッションが一定のメモリーを占める場合、セッションの一部をファイルに保存します。この作業は一定周期で実行されるモニタリング・スレッドによって実行されます。直列化を実行した結果のバイト単位の設定であり、-1または0を指定すると、実行しません</li> </ul>



# 索引

## A

Allow Fail Back, 51

## B

Backup Level, 51

Backup Queue Size, 51

Backup Unit Size, 51

BASE64, 14

BASIC

Cluster Mode, 40

Domain wide properties, 41

Exclude Das in Cluster, 41

Session Manager Provider, 41

Sticky Encoding Rule, 41

## C

Connect Timeout, 51

## F

Failover Delay, 52

## I

Ignore Flow Control, 52

## J

Jeus Login Manager, 52

Secondary, 52

Jeus Login Manager; primary, 52

## M

Max Session Count, 16

mod\_jkモジュール, 9

## P

Passivation, 52

Min Hole, 53

Packing Rate, 53

Path, 52

Ratio, 53

Single Folder File Limit, 53

Trigger, 53

Prevent Migration, 52

Properties, 52

## R

RAW, 14

Read Timeout, 51

Reserved Thread Num, 51

Restart Delay, 52

## S

Session Config, 17

Reload Persistent, 17

Session Cookie, 18

Comment, 20

Cookie Name, 18

Domain, 19

Http Only, 20

Max Age, 19

Path, 19

Secure, 19

Version, 19

Shared, 16

## T

Timeout, 15

Tracking Mode, 18

## U

URLリライティング, 3

## か

混合モード, 12

## さ

セッション・クッキー, 3

- セッション・サーバー, 9
- セッション・トラッキング, 1
- セッション・トラッキング・モード
  - SSL使用モード, 13
  - URLモード, 13
  - クッキー使用モード, 13
- セッション・ルーティング, 6

## は

- 分散型セッション・サーバー, 12
- 分散セッション・サーバーの構造
  - Backup Store, 26
  - Dynamic Remote Web Engine, 26
  - Session Manager, 25
  - SessionPacketHandler, 27