



目次

- 改訂情報
- はじめに
 - 本書の目的
 - 前提条件
 - 対象読者
 - 用語解説
- 各種インストール・設定変更
- intra-mart Accel Platform 構成ファイルの作成
- Payara のインストール
 - Payara をインストールする
 - Payara の起動と停止
 - 管理コンソールにアクセス
- Payara の設定
 - インスタンス設定の追加
 - JVM の設定
 - Network の設定
 - Thread Pool の設定
 - インスタンスの追加
 - データベースの接続設定
 - war ファイルのデプロイ
 - インスタンスの起動と停止
 - Executor Service の設定
- テナント環境の構築
- 付録
 - Payara のサービス化
 - NIC が複数ある場合
 - Payara のクラスタリング
 - Apache Solr
- セットアップで困ったら・・・
- アップデート パッチの適用

変更年月日	変更内容
2018-08-01	初版
2018-12-01	第2版 下記を変更しました 「Oracle の Connection Pool の作成」の「Resource Type」の説明を変更
2019-04-01	第3版 下記を変更しました 「intra-mart Accel Platform 構成ファイルの作成」に Metro と OpenPortal WSRP の選択に関する注意事項を追加
2020-04-01	第4版 下記を変更しました 「はじめに」の前提条件を変更 「JVM の設定」の「JVM Options の設定」の設定例を変更 「JVM の設定」の「JVM Options の設定」に注意書きを追加
2022-06-01	第5版 下記を変更しました 「Payara をインストールする」に jakarta.mail.jar の差し替えに関する記載を追加
2022-12-01	第6版 下記を変更しました 「SQL Server の Connection Pool の作成」に Additional Properties の設定で SendTimeAsDatetime を指定する手順を追加
2023-04-01	第7版 下記を変更しました 「JVM の設定」の「JVM Options の設定」に、カスタマーサクセスライセンスを利用する場合に必要な JVM 引数を追加のコラムを追加
2023-10-01	第8版 下記を変更しました 「はじめに」を Payara Server 5.2022.5 向けに変更 「Payara をインストールする」の「jakarta.mail.jar の差し替え」を削除 「JVM の設定」の「JVM Options の設定」に <code>--add-opens</code> を指定する手順を追加 「war ファイルのデプロイ」にデプロイに失敗した場合のコラムを追加

本書の目的

本書では Payara Server 5.2022.5 に intra-mart Accel Platform のセットアップを行う手順について説明します。



コラム

Payara についての詳しい情報は、Payara の Web サイトでご確認ください。

<https://www.payara.fish/> (English)

また、Payara Server のドキュメントも併せてご活用ください。

<https://docs.payara.fish/> (English)

前提条件

リリースノートに記載されているシステム要件を満たしている必要があります。

詳細は「[リリースノート](#)」を参照してください。

対象読者

以下の利用者を対象としています。

- Payara Server 5.2022.5 に intra-mart Accel Platform のセットアップを行われる方

用語解説

Payara のホームディレクトリを %PAYARA_HOME% と略します。

intra-mart Accel Platform のセットアップに必要なコンポーネントのインストールおよび設定を行います。

具体的な手順は「[intra-mart Accel Platform セットアップガイド](#)」の「[intra-mart Accel Platform を利用するためのミドルウェアのインストールと設定](#)」を参照してください。

— intra-mart Accel Platform セットアップガイド (Payara編) 第8版 2023-10-01

intra-mart Accel Platform 構成ファイルの作成

intra-mart Accel Platform の設定およびwarファイルの出力を行います。

具体的な手順は「[intra-mart Accel Platform セットアップガイド](#)」の「[WARファイルの作成](#)」を参照してください。

コラム

Payara Server 5.2022.5 で intra-mart Accel Platform を動作させるには、ベースモジュールとして 2023 Autumn(Hollyhock) 以降を選択する必要があります。

注意

追加リソースとして、必ず [Payara 5用設定ファイル] を出力する必要があります。

① 利用するプラットフォーム用の追加リソース(設定ファイル等)を配置します。

☐	Resin 4.0.x用設定ファイル
☒	Payara 5用設定ファイル
☐	Weblogic 12c用設定ファイル
☐	SAStruts用設定ファイル (Resin)
☐	SAStruts用設定ファイル (Payara)
☐	SAStruts用設定ファイル (WebSphere)
☐	SAStruts用設定ファイル (Weblogic)

また、SAStruts を利用する場合は、[SAStruts用設定ファイル (Payara)] を出力する必要があります。

① 利用するプラットフォーム用の追加リソース(設定ファイル等)を配置します。

☐	Resin 4.0.x用設定ファイル
☒	Payara 5用設定ファイル
☐	Weblogic 12c用設定ファイル
☐	SAStruts用設定ファイル (Resin)
☒	SAStruts用設定ファイル (Payara)
☐	SAStruts用設定ファイル (WebSphere)
☐	SAStruts用設定ファイル (Weblogic)

注意

Juggling プロジェクトのビルド時には、サーバ製品として必ず [Payara 5] を選択する必要があります。

サーバ製品の選択
対象となるサーバ製品を選択してください。

検索:

- ▲ Juggling
 - ▲ WAR file
 - ▲ Resin
 - ▲ Resin 4.0.x
 - ▲ Payara
 - ▲ Payara 5
 - ▲ Oracle WebLogic Server
 - ▲ Oracle WebLogic Server 12c
 - ▲ IBM WebSphere Application Server
 - ▲ WebSphere Application Server V8
 - 静的ファイル出力
 - モジュール構成Excel出力

注意

Payara Server を利用する場合、下記のモジュールを選択しないでください。デプロイ時にエラーが発生します。

- ライブラリ > サードパーティ製ライブラリ > Metro
- ライブラリ > サードパーティ製ライブラリ > OpenPortal WSRP

Payara をインストールする

項目

- インストール
- JDBC ドライバの配置

インストール

1. 以下から Full version の Payara Server 5.2022.5 をダウンロードします。
<https://www.payara.fish/downloads> (English)
2. ダウンロードした <payara-5.xxx.zip> を任意のディレクトリに展開します。

JDBC ドライバの配置

1. JDBC ドライバをダウンロードします。

JDBC ドライバの入手先は以下を参照してください。

[JDBCドライバ](#)

2. ダウンロードした、JDBC ドライバを以下のディレクトリにコピーします。

- Linux

```
%PAYARA_HOME%/glassfish/lib
```

- Windows

```
%PAYARA_HOME%\glassfish\lib
```

Payara の起動と停止

項目

- サーバの起動
- サーバの停止
- サーバの再起動
- サーバのログ

サーバの起動

1. 以下のコマンドを実行します。

```
%PAYARA_HOME%/glassfish/bin/asadmin start-domain
```

または

```
%PAYARA_HOME%/glassfish/bin/startserv
```

2. 起動が完了すると以下のメッセージが出力されます。

- `asadmin start-domain` で起動した場合

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

- `startserv` で起動した場合

```
[#|2023-05-15T15:17:42.913+0900|情報|Payara 5.2022.5|javax.enterprise.web|_ThreadID=151;_ThreadName=Thread-13;_TimeMillis=1684131462913;_LevelValue=800;_MessageID=AS-WEB-GLUE-00172;|
Loading application [_admingui] at [/]|#]
```

サーバの停止

1. 以下のコマンドを実行します。

```
%PAYARA_HOME%/glassfish/bin/asadmin stop-domain
```

または

```
%PAYARA_HOME%/glassfish/bin/stopserv
```

2. 停止が完了すると以下のメッセージが出力されます。

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

サーバの再起動

1. 以下のコマンドを実行します。

```
%PAYARA_HOME%/glassfish/bin/asadmin restart-domain
```

2. 再起動が完了すると以下のメッセージが出力されます。

```
Command restart-domain executed successfully.
```

サーバのログ

以下のファイルに出力されます。

```
%PAYARA_HOME%/glassfish/domains/domain1/logs/server.log
```

管理コンソールにアクセス

Payara をインストールしたホスト（ローカル）から管理コンソールにアクセスする場合は、特別な設定なしにブラウザからアクセスできます。

別のホストからアクセスする場合は、セキュア管理を有効にする必要があります。

項目

- [ローカルから管理コンソールにアクセス](#)
- [リモートホストから管理コンソールにアクセス](#)

ローカルから管理コンソールにアクセス

1. サーバを起動します。起動方法は「[サーバの起動](#)」を参照してください。
2. ブラウザで以下にアクセスします。

```
http://localhost:4848
```

リモートホストから管理コンソールにアクセス

1. サーバを起動します。起動方法は「[サーバの起動](#)」を参照してください。
2. 以下のコマンドを実行して管理者のパスワードを設定します。

- Linux

```
%PAYARA_HOME%\glassfish\bin\asadmin change-admin-password
```

- Windows

```
%PAYARA_HOME%\glassfish\bin\asadmin.bat change-admin-password
```

上記のコマンドを実行すると、以下の入力を求められます。

Enter admin user name [default: admin]> 管理者のユーザ名を入力します。デフォルトは admin です。

Enter the admin password> 管理者のパスワードを入力します。デフォルトは 空 です。

Enter the new admin password> 管理者の新しいパスワードを入力します。

Enter the new admin password again> 管理者の新しいパスワードを再度入力します。

3. 以下のコマンドを実行して、セキュア管理を有効にします。

- Linux

```
%PAYARA_HOME%\glassfish\bin\asadmin enable-secure-admin
```

- Windows

```
%PAYARA_HOME%\glassfish\bin\asadmin.bat enable-secure-admin
```

上記のコマンドを実行すると、以下の入力を求められます。

Enter admin password for user "admin"> 管理者のパスワードを入力します。

4. サーバ再起動します。再起動方法は「[サーバの再起動](#)」を参照してください。

5. ブラウザで以下にアクセスします。

```
https://<%PAYARA_HOST%>:4848
```

Payara の設定を行います。

コラム

インスタンスを作成することで独立した JVM で intra-mart Accel Platform を構築できます。
これにより、デプロイした別のアプリケーションの影響を受けずに intra-mart Accel Platform を稼働できます。

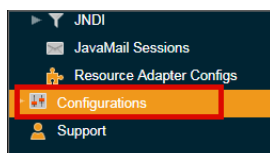
ここでは、インスタンスを作成する方法も含めて、Payara の設定方法をご紹介します。

注意

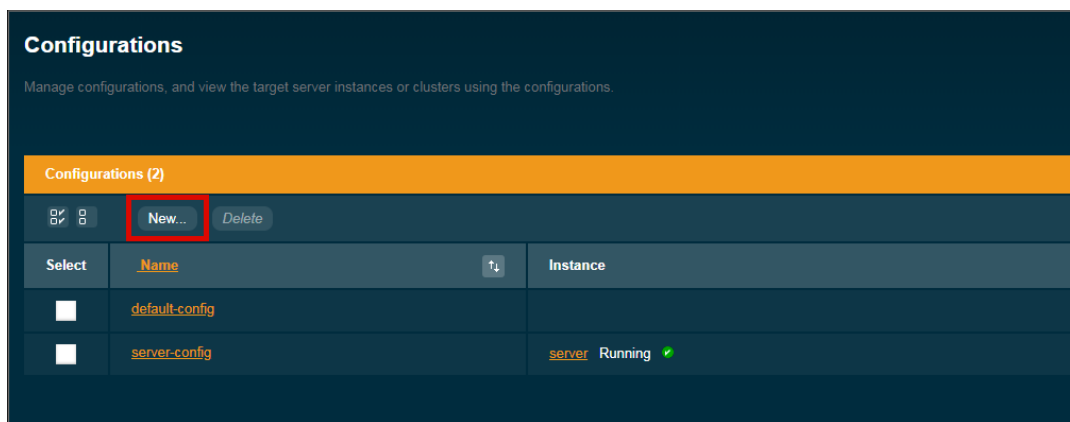
インスタンスを作成しない場合、[Availability Service] は利用できません。
これにより、セッションレプリケーションなどを利用できません。
分散環境を構築する場合は、必ずインスタンスを作成してください。

インスタンス設定の追加

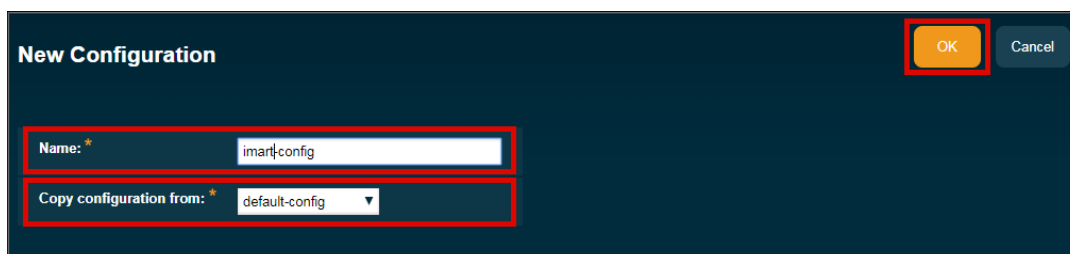
1. Payara の管理コンソールにアクセスします。アクセス方法は「[管理コンソールにアクセス](#)」を参照してください。
2. 左ペインのメニューから [Configurations] を選択します。



3. [New...] を選択します。



4. [New Configuration] で以下を設定して [OK] を選択します。



New Configuration

Name: *

Copy configuration from: *

Name 設定の名前を入力します。例として、ここでは imart-config と入力します。

Copy configuration from コピー元の設定の名前を選択します。例として、ここでは default-config を選択します。

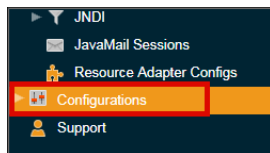
JVM の設定

項目

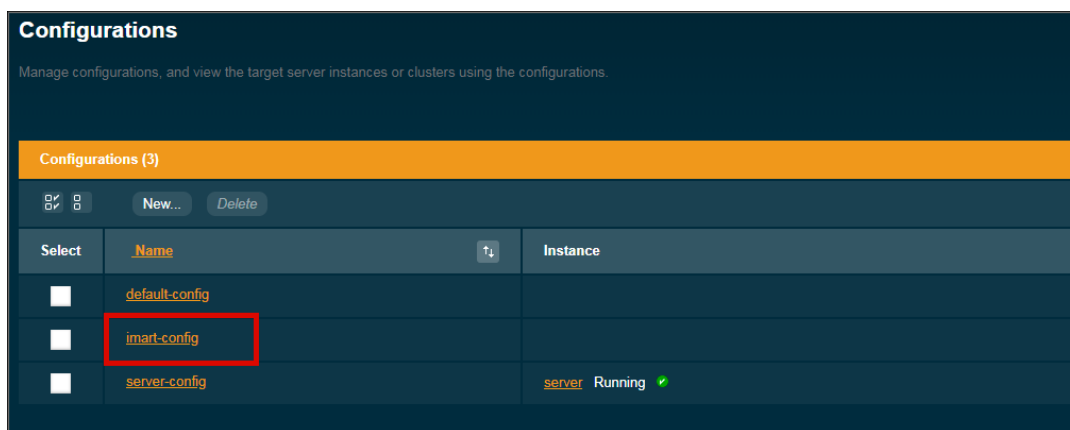
- JVM Options の設定
- Debug の設定

JVM Options の設定

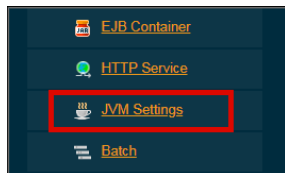
1. Payara の管理コンソールにアクセスします。アクセス方法は「[管理コンソールにアクセス](#)」を参照してください。
2. 左ペインのメニューから [Configurations] を選択します。



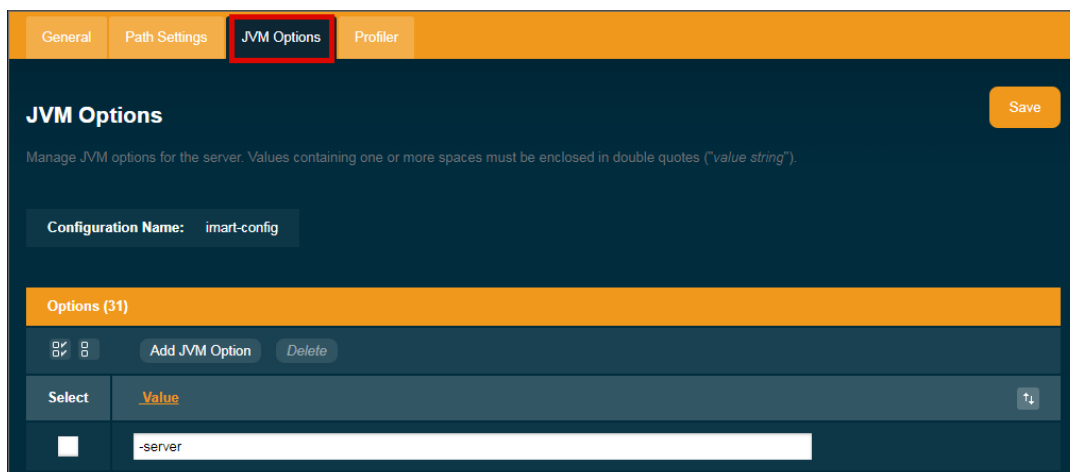
3. 「[インスタンス設定の追加](#)」で追加した設定を選択します。例として、ここでは [imart-config] を選択します。



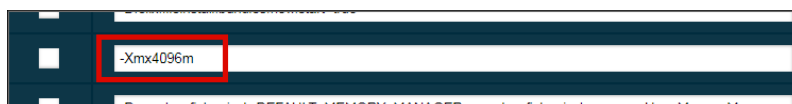
4. [JVM Settings] を選択します。



5. [JVM Options] を選択します。



6. Java 最大ヒープサイズの設定を行います。デフォルトで設定されている値から以下のように変更します。



この例では、以下を設定しています。

-Xmx4096m

7. [Add JVM Option] をクリックして、`--add-opens` オプションを設定します。

☐			<code>--add-opens=java.base/sun.util.locale=ALL-UNNAMED</code>
☐			<code>--add-opens=java.xml/com.sun.org.apache.xerces.internal.jaxp=ALL-UNNAMED</code>
☐			<code>--add-opens=java.base/sun.util.calendar=ALL-UNNAMED</code>
☐			<code>--add-opens=java.xml/com.sun.org.apache.xerces.internal.jaxp.datatype=ALL-UNNAMED</code>
☐			<code>--add-opens=java.base/java.text=ALL-UNNAMED</code>
☐			<code>--add-opens=java.desktop/java.awt.font=ALL-UNNAMED</code>
☐			<code>--add-opens=java.base/java.math=ALL-UNNAMED</code>
☐			<code>--add-opens=java.base/java.util.concurrent=ALL-UNNAMED</code>

この例では、以下を設定しています。

```
--add-opens=java.base/sun.util.locale=ALL-UNNAMED
--add-opens=java.xml/com.sun.org.apache.xerces.internal.jaxp=ALL-UNNAMED
--add-opens=java.base/sun.util.calendar=ALL-UNNAMED
--add-opens=java.xml/com.sun.org.apache.xerces.internal.jaxp.datatype=ALL-UNNAMED
--add-opens=java.base/java.text=ALL-UNNAMED
--add-opens=java.desktop/java.awt.font=ALL-UNNAMED
--add-opens=java.base/java.math=ALL-UNNAMED
--add-opens=java.base/java.util.concurrent=ALL-UNNAMED
```

8. [Add JVM Option] をクリックして、その他の JVM Option を設定します。

Options (63)			
Select	Min JVM Version	Max JVM Version	Value
☐			<code>-Xms4096m</code>
☐			<code>-Dfile.encoding=UTF-8</code>
☐			<code>-Duser.timezone=Asia/Tokyo</code>
☐			<code>-XX:+UseG1GC</code>
☐			<code>-XX:MaxGCPauseMillis=200</code>
☐			<code>-XX:InitiatingHeapOccupancyPercent=30</code>
☐			<code>-XX:+HeapDumpOnOutOfMemoryError</code>
☐			<code>-Djava.io.tmpdir=/var/payara-tmp</code>

この例では、以下を設定しています。

```
-Xms4096m
-Dfile.encoding=UTF-8
-Duser.timezone=Asia/Tokyo
-XX:+UseG1GC
-XX:MaxGCPauseMillis=200
-XX:InitiatingHeapOccupancyPercent=30
-XX:+HeapDumpOnOutOfMemoryError
-Djava.io.tmpdir=/var/payara-tmp
--add-opens=java.base/sun.util.locale=ALL-UNNAMED
--add-opens=java.xml/com.sun.org.apache.xerces.internal.jaxp=ALL-UNNAMED
--add-opens=java.base/sun.util.calendar=ALL-UNNAMED
--add-opens=java.xml/com.sun.org.apache.xerces.internal.jaxp.datatype=ALL-UNNAMED
--add-opens=java.base/java.text=ALL-UNNAMED
--add-opens=java.desktop/java.awt.font=ALL-UNNAMED
--add-opens=java.base/java.math=ALL-UNNAMED
--add-opens=java.base/java.util.concurrent=ALL-UNNAMED
```



コラム

「-Djava.io.tmpdir」オプションを追加し、Payara Server が利用する作業ディレクトリを変更できます。このオプションが指定されていない場合、Payara Server が利用する作業ディレクトリは JVM のデフォルトの設定が利用されます。



コラム

カスタマーサクセスライセンスをご契約中の場合には、ライセンスポータルと通信してご契約内容の変更が自動反映されます。

Payara をインストールした環境からライセンスポータルへの通信にプロキシサーバを利用する場合は、次の JVM 引数の設定が必要です。

-Dhttps.proxyHost	プロキシサーバのホストURL
-Dhttps.proxyPort	プロキシサーバのポート番号
-Dhttps.proxyUser	プロキシサーバへの接続ユーザ
-Dhttps.proxyPassword	接続ユーザのパスワード

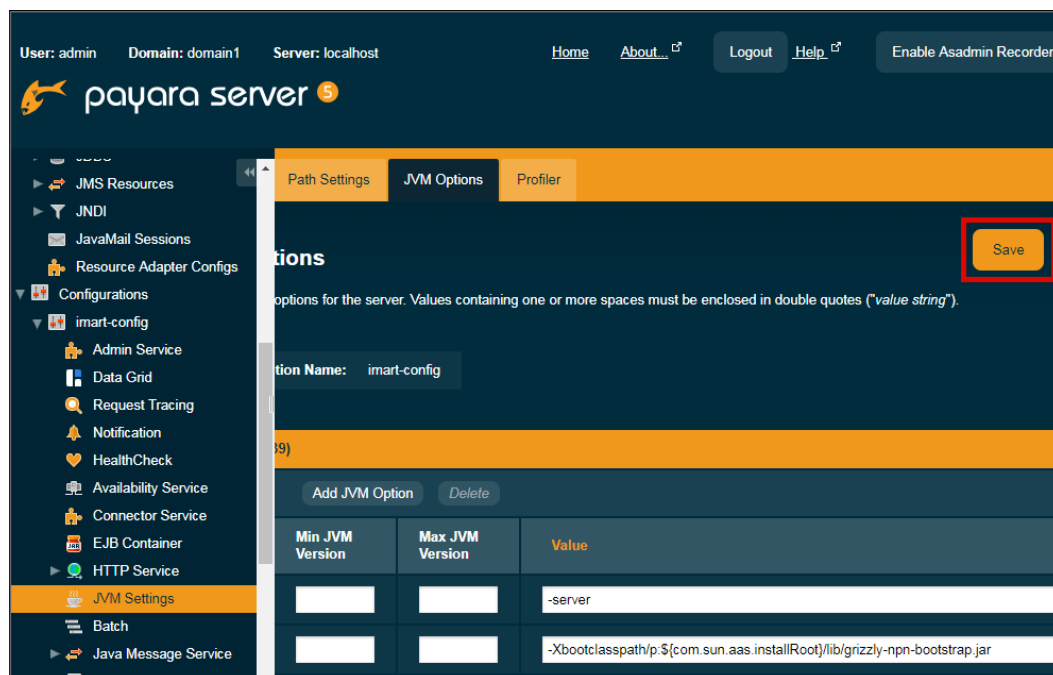
カスタマーサクセスライセンスについての詳細は、「[ライセンスの登録](#)」を参照してください。



注意

- 「-Djava.io.tmpdir」オプションにより指定されたディレクトリは事前に作成しておく必要があります。
- Payara Server 実行ユーザが読み込み、書き込みを行うことができる権限を設定しておく必要があります。
- Linux 系の環境では、このオプションが未指定の場合 /tmp が利用されます。cron 等の設定により定期的に/tmp 配下の内容が削除される設定が標準で組み込まれている場合があります。

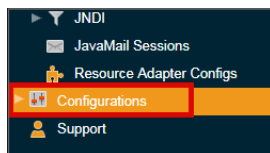
9. [Save] を選択します。



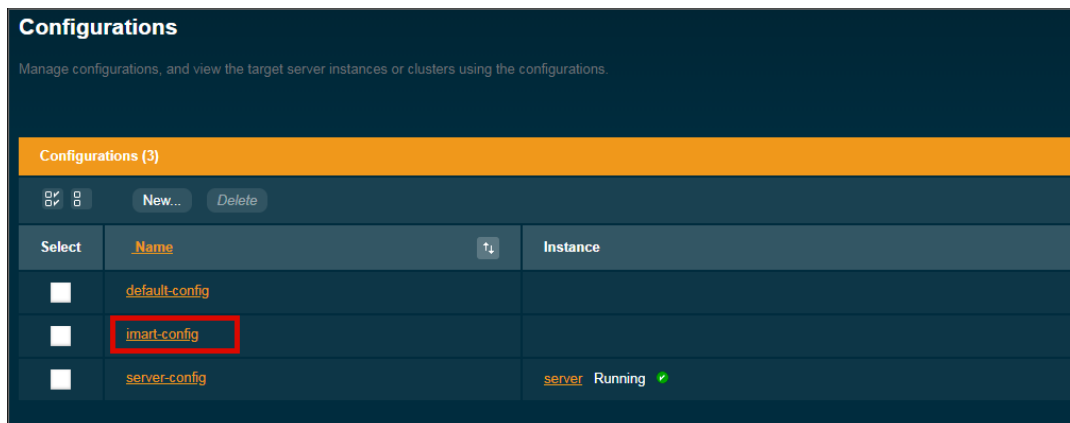
Debug の設定

Java のリモートデバッグを利用する場合、以下の手順を実施してください。

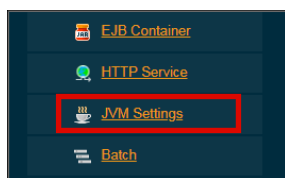
1. Payara の管理コンソールにアクセスします。アクセス方法は「[管理コンソールにアクセス](#)」を参照してください。
2. 左ペインのメニューから [Configurations] を選択します。



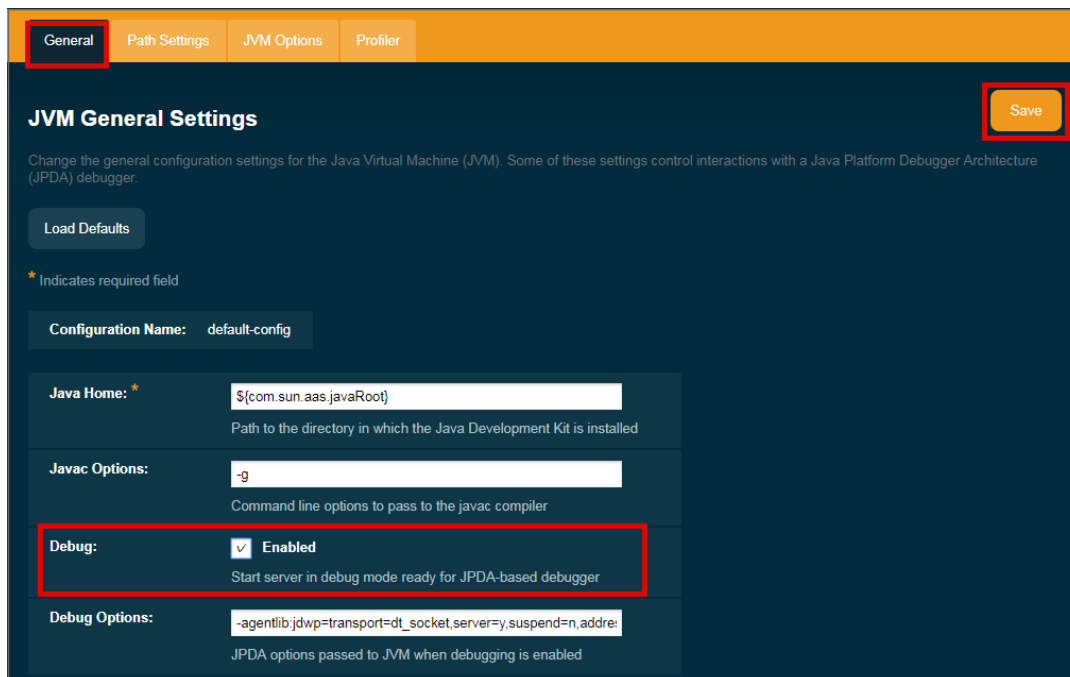
3. 「[インスタンス設定の追加](#)」で追加した設定を選択します。例として、ここでは [imart-config] を選択します。



4. [JVM Settings] を選択します。



5. [General] で [Debug] を ON にして [Save] を選択します。



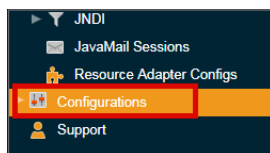
Network の設定

項目

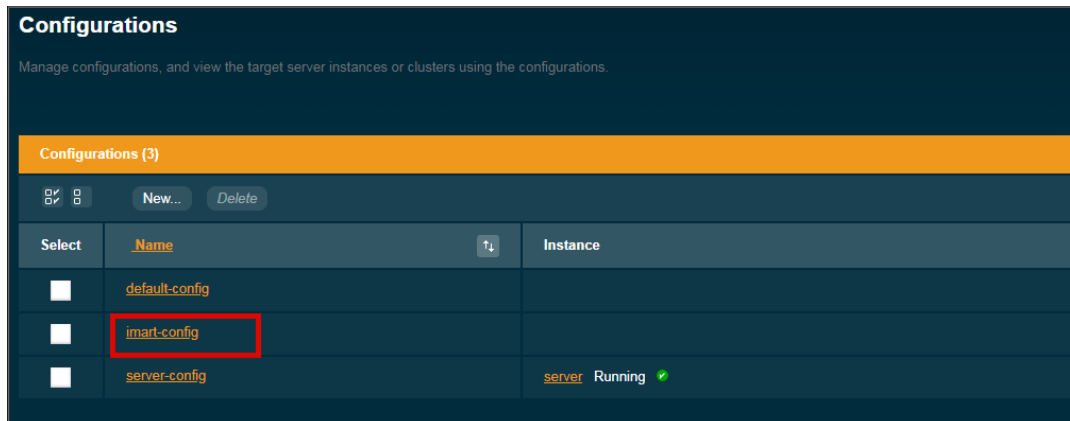
- [Network Listener の設定](#)

Network Listener の設定

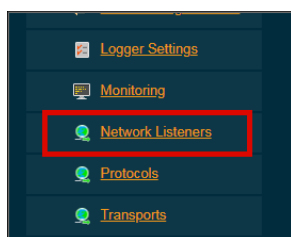
1. Payara の管理コンソールにアクセスします。アクセス方法は「[管理コンソールにアクセス](#)」を参照してください。
2. 左ペインのメニューから [Configurations] を選択します。



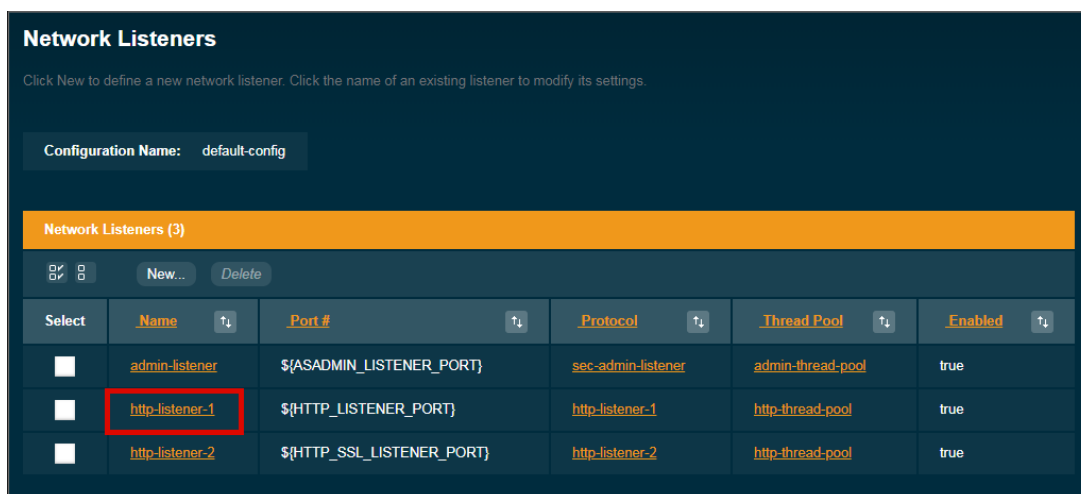
3. 「[インスタンス設定の追加](#)」で追加した設定を選択します。例として、ここでは [imart-config] を選択します。



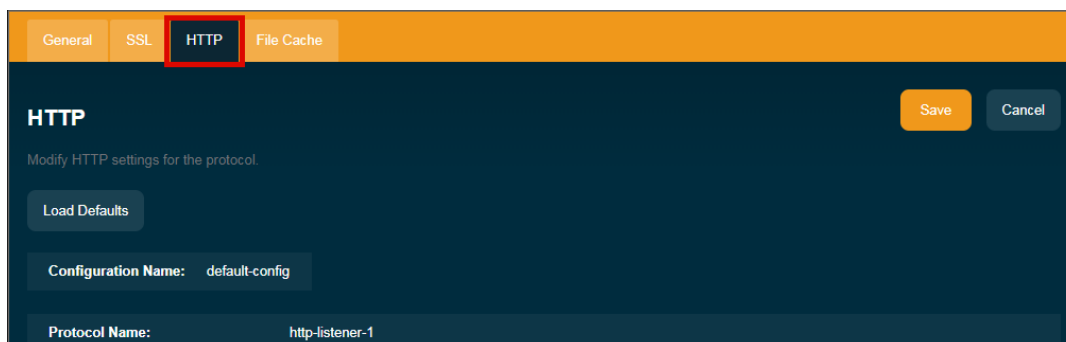
4. [Network Listeners] を選択します。



5. [http-listener-1] を選択します。



6. [HTTP] を選択します。



7. [Comet Support] と [Encoded Slash] を ON に設定して [Save] 選択します。



Modify HTTP settings for the protocol.

[Load Defaults](#)

Configuration Name: default-config

Protocol Name:	http-listener-1	
Server Name:		
	Alias name if server uses an alias. If a colon and port number are appended, that port will be used in URLs the server sends to the client.	
Default Virtual Server:	server ▼	
	Use the Virtual Servers page to define a new virtual server	
Redirect Port:		
	SSL port value for redirects	
Max Connections:	256	
	Maximum number of requests per connection in keep-alive mode.	
Timeout:	30	Seconds
	Maximum time a connection can be deemed as idle and kept in the keep-alive state. A value of -1 will disable it.	
Upload Timeout:	☒ Enabled	
	Enable closing of connection for a servlet that reads bytes slowly after Connection Upload Timeout is reached	
Connection Upload Timeout:	300000	Milliseconds
	Timeout for uploads if Upload Timeout is enabled. A value of -1 will disable it.	
Request Timeout:	900	Seconds
	Time after which a request times out. A value of -1 will disable it.	
Send Buffer Size:	8192	Bytes
	Size of the send buffer	
Header Buffer Length:	8192	Bytes
	The size of the buffer used by the request processing threads to read the request data	
Max Post Size:	-1	Bytes
	Maximum size of POST actions	
Max Form Post Size:	2097152	Bytes
	Maximum size of a POST form	
Max Save Post Size	4096	Bytes
	Maximum size of a POST which will be saved by the container during authentication.	
URI Encoding:	UTF-8	
	Character set used to decode the request URIs received	
HTTP/2:	☒ Enabled	
	Enable HTTP/2.	
Disable HTTP/2 Cipher Check:	☐ Enabled	
	Whether or not insecure cipher suites are allowed to establish TLS connections.	
HTTP/2 Max Concurrent Streams:	100	
	The number of concurrent streams allowed per HTTP/2 connection. The default is 100.	
HTTP/2 Initial Window Size:	65535	Bytes
	The initial window size in bytes. The default is 64K - 1.	
HTTP/2 Max Frame Payload Size:	16777215	Bytes
	The maximum size of the HTTP2 frame payload to be accepted. The default is 2^24 - 1.	
HTTP/2 Max Header List Size:	4096	Bytes
	The maximum size, in bytes, of the header list.	
Compression:	off ▼	
	Enable HTTP/1.1 GZIP compression to save server bandwidth	
Compressible Mime Types:	text/html,text/xml,text/plain	
	Comma-separated list of MIME types for which HTTP compression is used	
Compression Minimum Size:	2048	Bytes
	Minimum size of a file when compression is applied	
No-Compression User Agents:		
	Comma-separated list of regular expressions matching user agents of HTTP clients for which compression should not be used	

Restricted User Agent:		List of restricted user agents on which HTTP compression is applied
Default Response Type:		Specified as a semi-colon delimited string consisting of content-type, encoding, language, charset
Forced Response Type:		The response type to be forced if the content served cannot be matched by any of the MIME mappings for extensions
Adapter:	org.glassfish.grizzly.http.server.StaticHttpHandler	Class name of the static resources adapter
Comet Support:	☒ Enabled	Enable Comet support
DNS Lookup:	☐ Enabled	Enable Domain Name System (DNS) lookup
RCM Support:	☐ Enabled	Enable Resource Configuration Management (RCM) support
Trace:	☐ Enabled	Enable TRACE operation
Auth Pass Through:	☐ Enabled	Indicate that the network listener receives traffic from an SSL-terminating proxy server
Chunking:	☒ Enabled	Enable HTTP response chunking
XPowered By:	☒	Include X-Powered-By: Servlet/3.0 in servlet-generated HTTP response headers
Server Header:	☒	Include Server Header: Servlet/3.0 in servlet-generated HTTP response headers
XFrame Options:	☒	Enabling this will set X-Frame-Options to "SAMEORIGIN" value in all HTTP response headers. If you have declared it in your application, it will take precedence.
Encoded Slash:	☒ Enabled	Allow encoded slash in URIs
Websockets Support:	☒ Enabled	
Scheme Mapping		HTTP Header Name used for identifying the originating protocol of an HTTP request
Remote User Mapping		HTTP Header Name used for identifying the originating user of an HTTP request

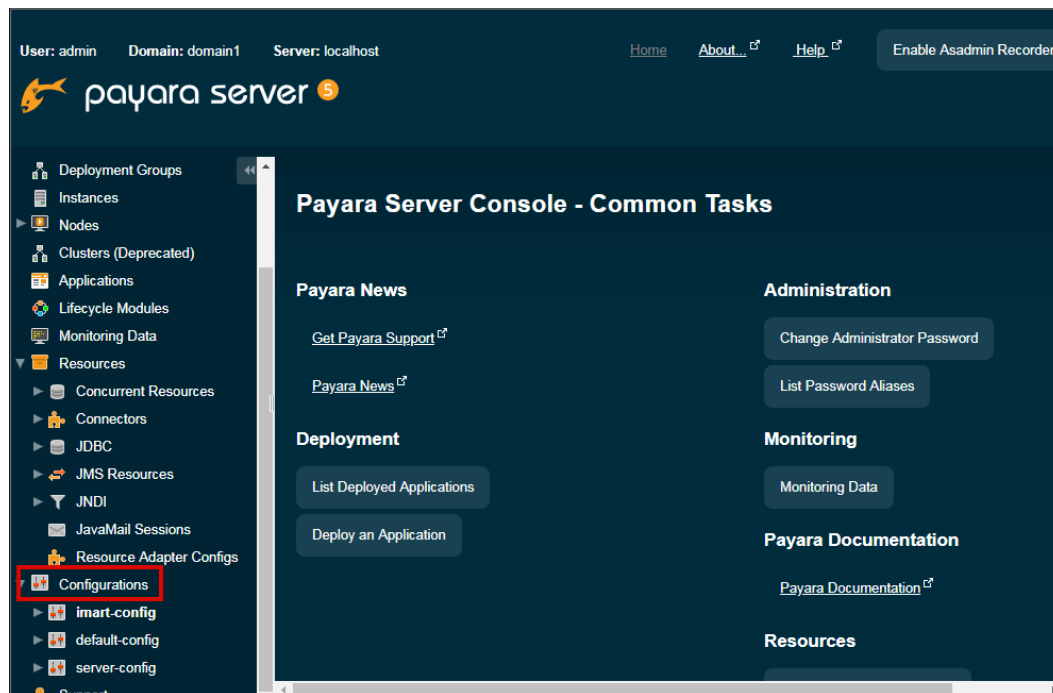
Thread Pool の設定

項目

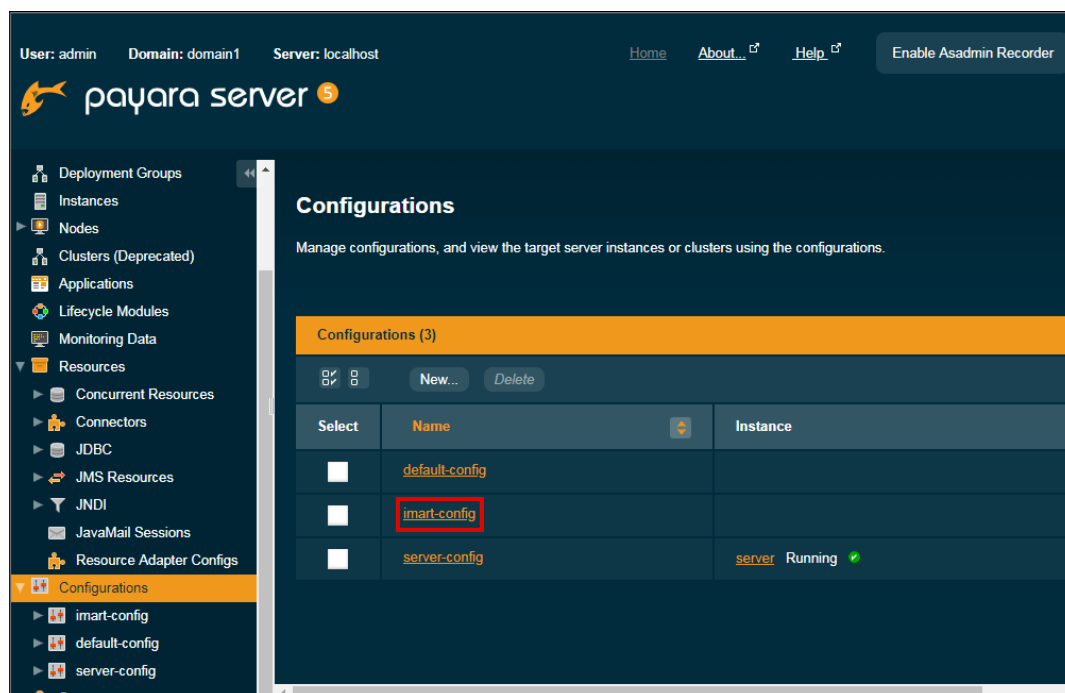
- Thread Pool の設定

Thread Pool の設定

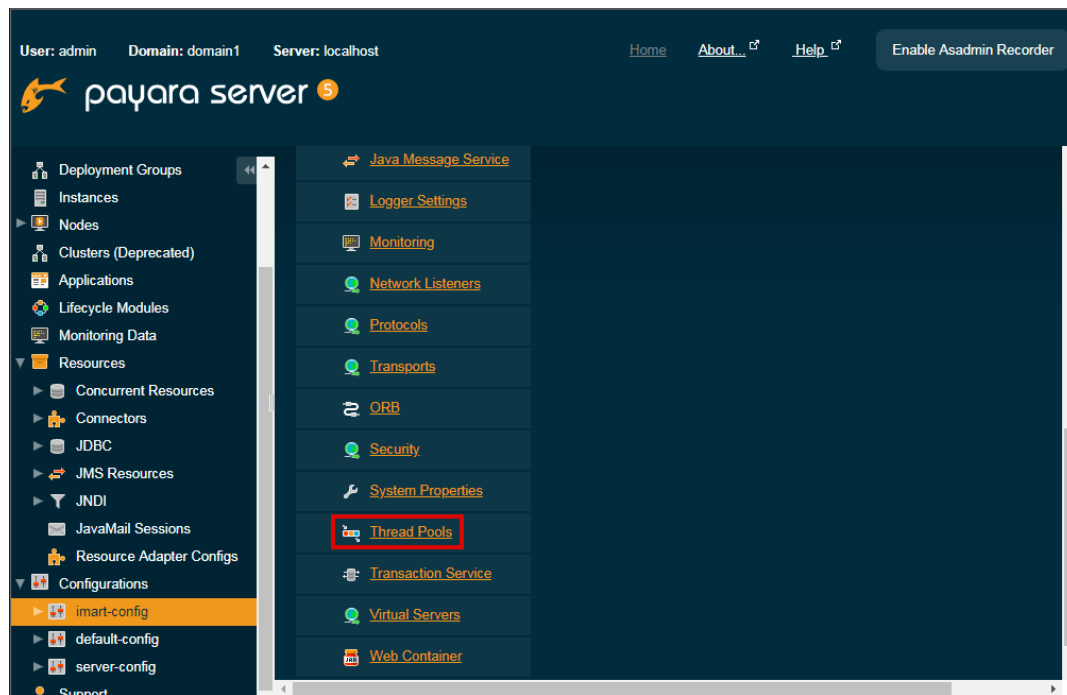
1. Payara の管理コンソールにアクセスします。アクセス方法は「[管理コンソールにアクセス](#)」を参照してください。
2. 左ペインのメニューから [Configurations] を選択します。



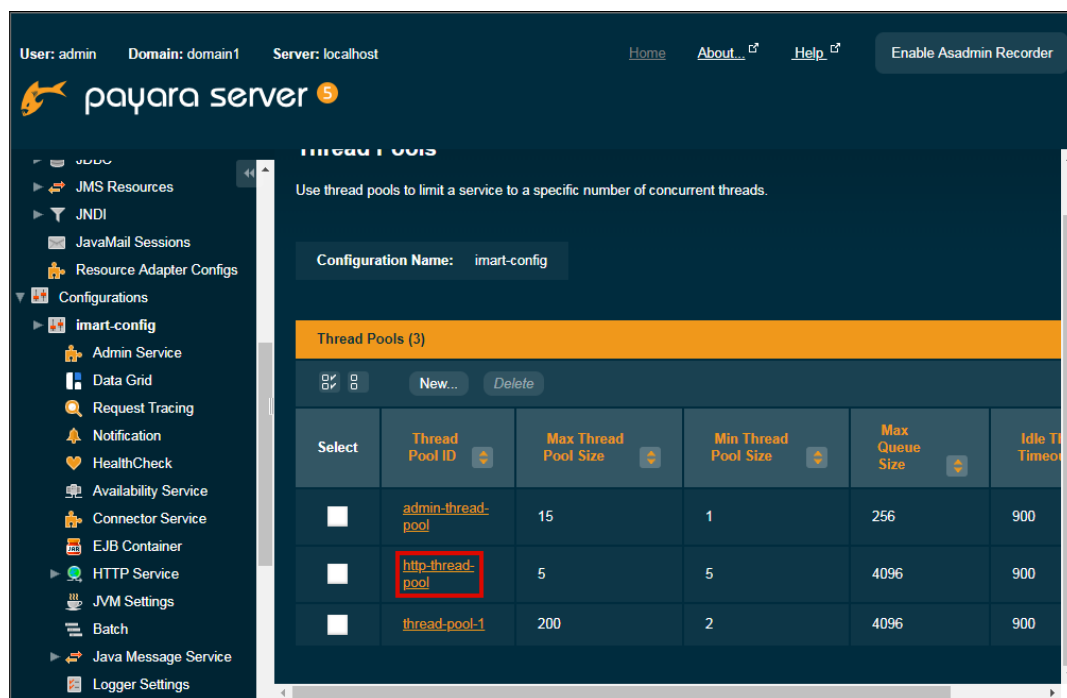
3. 「インスタンス設定の追加」で追加した設定を選択します。例として、ここでは [imart-config] を選択します。



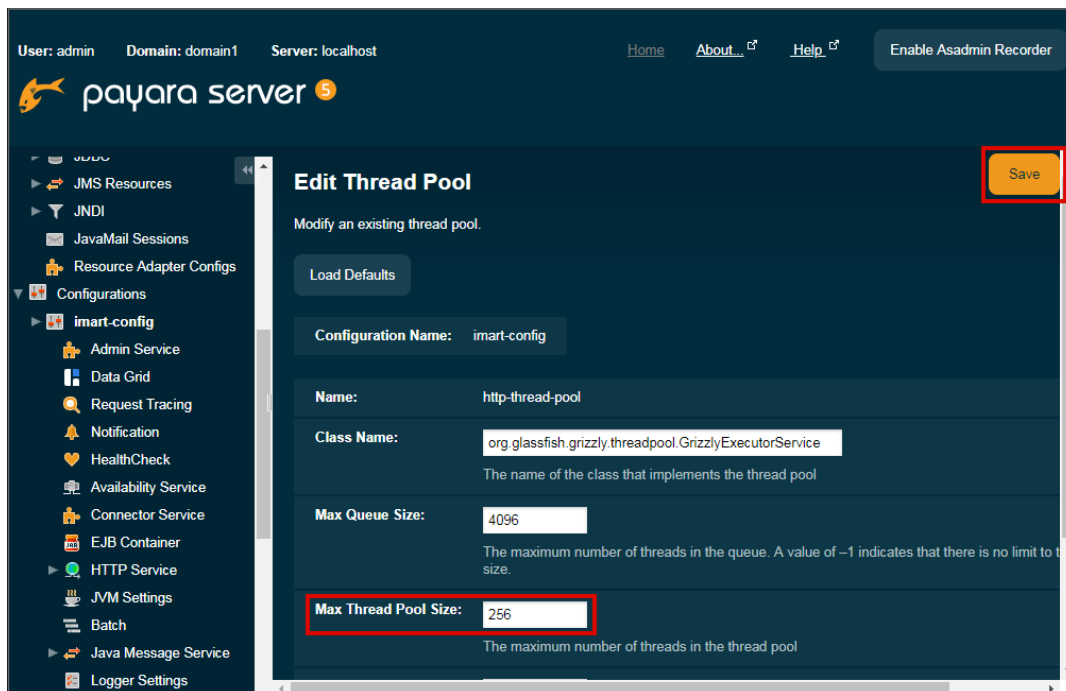
4. [Thread Pools] を選択します。



5. [http-thread-pool] を選択します。

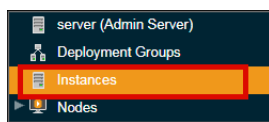


6. [Max Thread Pool Size] に 256 を設定して [Save] 選択します。

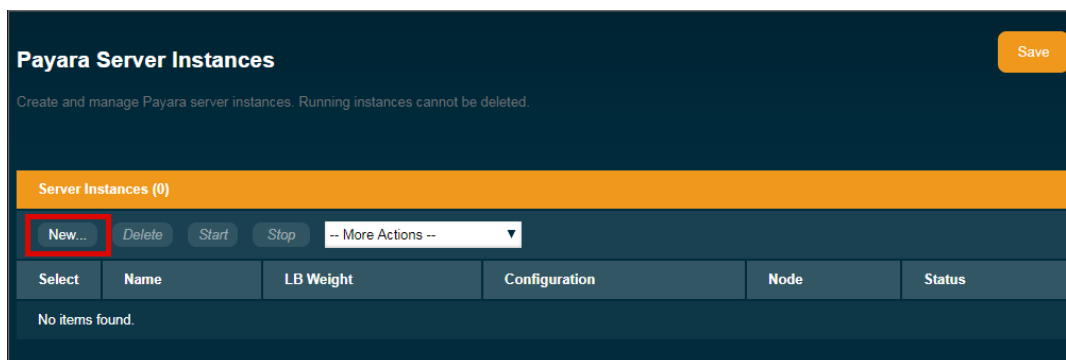


インスタンスの追加

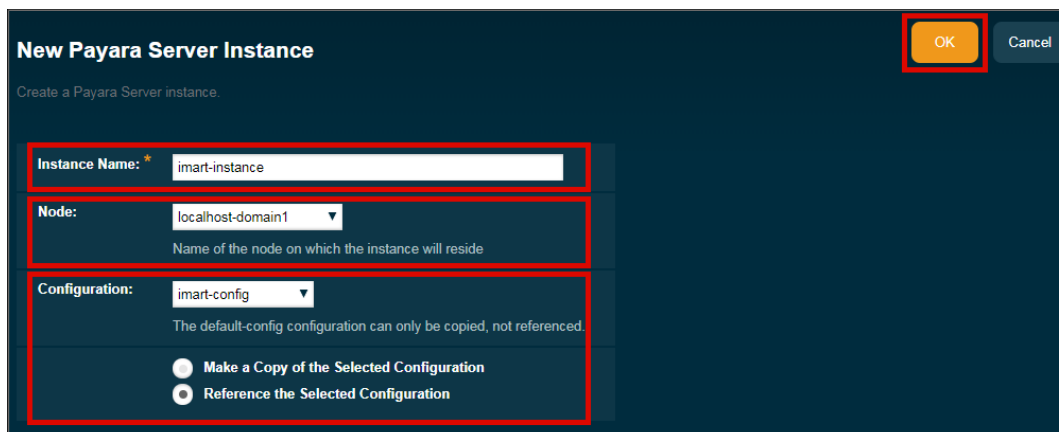
1. Payara の管理コンソールにアクセスします。アクセス方法は「[管理コンソールにアクセス](#)」を参照してください。
2. 左ペインのメニューから [Instances] を選択します。



3. [New...] を選択します。



4. [New Payara Server Instance] で以下のように入力して [OK] を選択します。



Instance Name インスタンスの名前を入力します。例として、ここでは imart-instance と入力します。

Node	ノードを選択します。例として、ここでは localhost-domain1 を選択します。
Configuration	「 インスタンス設定の追加 」で追加した設定を選択します。例として、ここでは imart-config を選択します。 また、「 インスタンス設定の追加 」で追加した設定をこのインスタンスの設定として利用するため、 [Reference the Selected Configuration] を選択します。

データベースの接続設定

データベースの接続設定を行います。

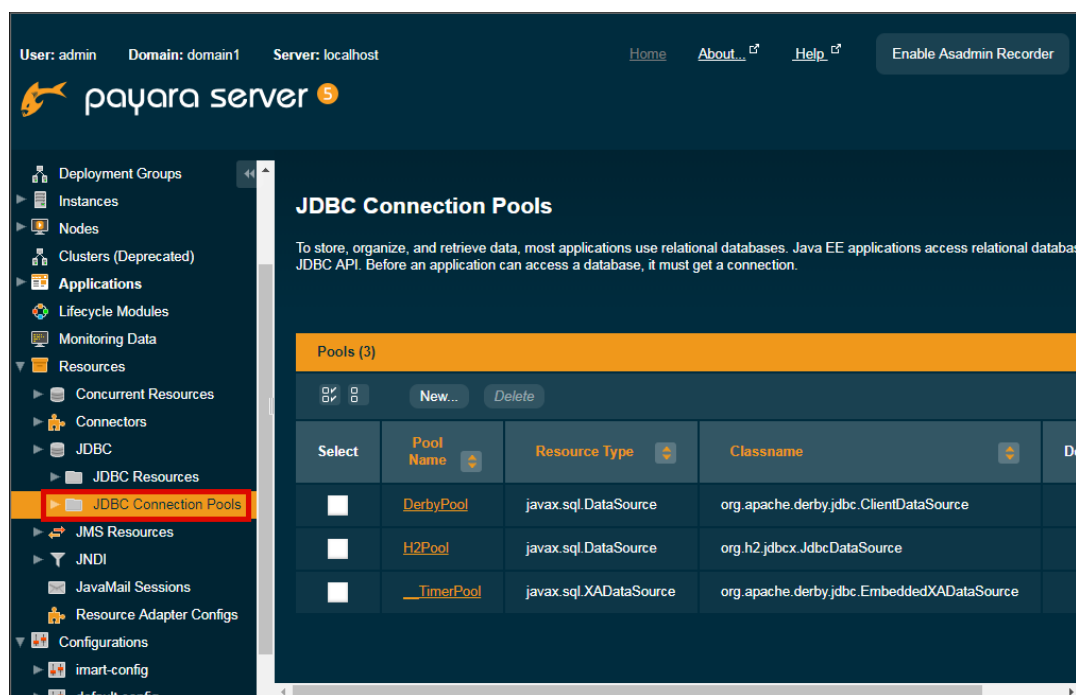
Connection Pool の作成

使用するデータベースに応じて設定を行ってください。

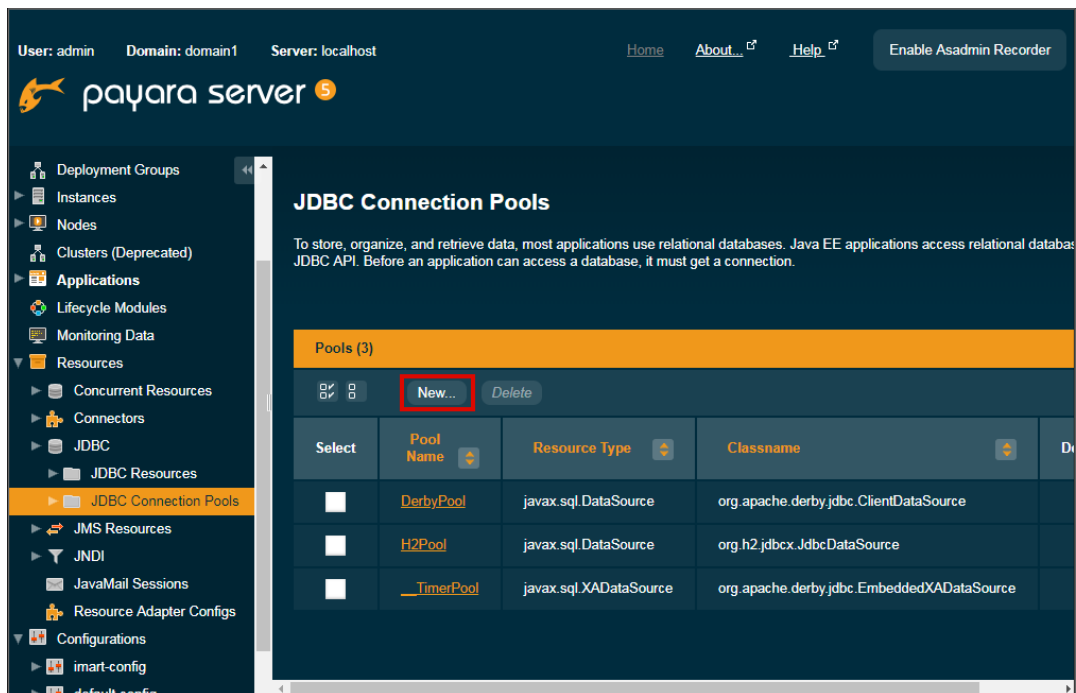
PostgreSQL の Connection Pool の作成

PostgreSQL を使用するための設定を行います。

1. Payara の管理コンソールにアクセスします。アクセス方法は「[管理コンソールにアクセス](#)」を参照してください。
2. 左ペインのメニューから [Resources] - [JDBC] - [JDBC Connection Pools] を選択します。



3. [New...] を選択します。



4. [New JDBC Connection Pool (Step 1 of 2)] で以下のように入力して [Next] を選択します。

New JDBC Connection Pool (Step 1 of 2)

Identify the general settings for the connection pool.

* Indicates required field

General Settings

Pool Name:

Resource Type:
Must be specified if the datasource class implements more than 1 of the interface.

Database Driver Vendor:
Select or enter a database driver vendor

Introspect: ☒ Enabled
If enabled, data source or driver implementation class names will enable introspection.

Next **Cancel**

Pool Name コネクションプールの名前を入力します。例として、ここでは PgPool と入力します。

Resource Type リソースタイプを選択します。 javax.sql.XADataSource を選択します。

Database Driver Vendor JDBC ドライバのベンダを選択します。 Postgresql を選択します。

5. [New JDBC Connection Pool (Step 2 of 2)] の [Additional Properties] で以下を変更します。

Additional Properties (58)

Select	Name	Value	Description
☐	Url	jdbc:postgresql://localhost/iap_db	
☐	User	imart	
☐	Password	imart	
☐	PreparedStatementCacheQueries	0	
☐	PreparedStatementCacheSizeMiB	0	

Url データベースURLを設定します。例として、ここでは `jdbc:postgresql://localhost/iap_db` を設定しています。

User	データベースユーザを設定します。例として、ここでは imart を設定しています。
Password	データベースユーザのパスワードを設定します。例として、ここでは imart を設定しています。
PreparedStatementCacheQueries	プリペアドステートメントのキャッシュ数を設定します。例として、ここでは 0 を設定しています。
PreparedStatementCacheSizeMiB	プリペアドステートメントのキャッシュサイズを設定します。例として、ここでは 0 を設定しています。

6. [New JDBC Connection Pool (Step 2 of 2)] で [Finish] を選択します。

New JDBC Connection Pool (Step 2 of 2)

Identify the general settings for the connection pool. Datasource Classname or Driver Classname must be specified for the connection pool.

* Indicates required field

General Settings

Pool Name: PgPool

Resource Type: javax.sql.XADataSource

Database Driver Vendor: PostgreSQL

Datasource Classname: org.postgresql.xa.PGXADataSource

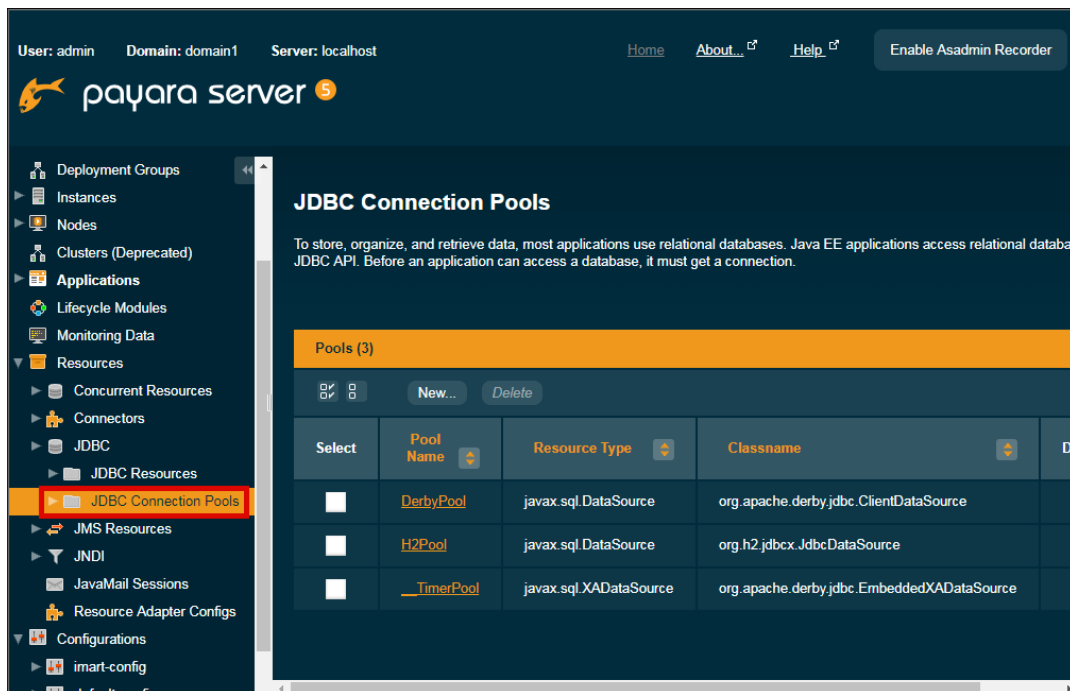
Select or enter vendor-specific classname that implements the DataSource and/or XADataSource APIs

Driver Classname:

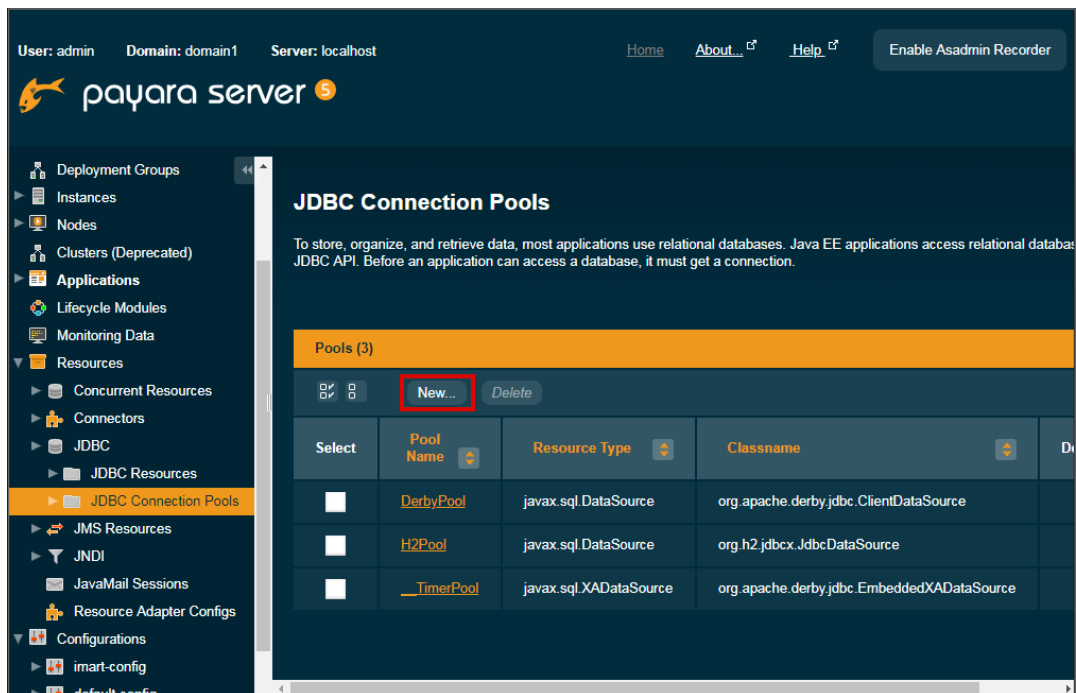
Oracle の Connection Pool の作成

Oracle を使用するための設定を行います。

1. Payara の管理コンソールにアクセスします。アクセス方法は「[管理コンソールにアクセス](#)」を参照してください。
2. 左ペインのメニューから [Resources] - [JDBC] - [JDBC Connection Pools] を選択します。



3. [New...] を選択します。



4. [New JDBC Connection Pool (Step 1 of 2)] で以下のように入力して [Next] を選択します。

The screenshot shows the 'New JDBC Connection Pool (Step 1 of 2)' form. The 'Next' button is highlighted with a red box. The form fields are also highlighted with red boxes.

General Settings

Pool Name: * OraclePool

Resource Type: javax.sql.XADataSource
Must be specified if the datasource class implements more than 1 of the interface.

Database Driver Vendor: Oracle
Select or enter a database driver vendor

Introspect: ☐ Enabled
If enabled, data source or driver implementation class names will enable introspection.

Pool Name コネクションプールの名前を入力します。例として、ここでは OraclePool と入力します。

Resource Type リソースタイプを選択します。 javax.sql.XADataSource を選択します。

Database Driver Vendor JDBC ドライバのベンダを選択します。 Oracle を選択します。

5. [New JDBC Connection Pool (Step 2 of 2)] の [Additional Properties] で以下を変更します。

The screenshot shows the 'Additional Properties (19)' table. The 'URL', 'User', and 'Password' rows are highlighted with a red box.

Select	Name	Value	Description
☐	URL	jdbc:oracle:thin:@localhost:1521:orcl	
☐	User	IMART	
☐	Password	imart	

URL データベースURLを設定します。例として、ここでは jdbc:oracle:thin:@localhost:1521:orcl を設定しています。

User データベースユーザを設定します。例として、ここでは IMART を設定しています。

Password

データベースユーザのパスワードを設定します。例として、ここでは imart を設定しています。

6. [New JDBC Connection Pool (Step 2 of 2)] で [Finish] を選択します。

New JDBC Connection Pool (Step 2 of 2)

Identify the general settings for the connection pool. Datasource Classname or Driver Classname must be specified for the connection pool.

* Indicates required field

General Settings

Pool Name: OraclePool

Resource Type: javax.sql.XADataSource

Database Driver Vendor: Oracle

Datasource Classname: oracle.jdbc.xa.client.OracleXADataSource

Select or enter vendor-specific classname that implements the DataSource and/or XADataSource APIs

Driver Classname:

SQL Server の Connection Pool の作成

SQL Server を使用するための設定を行います。

1. Payara の管理コンソールにアクセスします。アクセス方法は「[管理コンソールにアクセス](#)」を参照してください。
2. 左ペインのメニューから [Resources] - [JDBC] - [JDBC Connection Pools] を選択します。

User: admin Domain: domain1 Server: localhost

Home About... Help

Enable Asadmin Recorder

payara server

Deployment Groups

Instances

Nodes

Clusters (Deprecated)

Applications

Lifecycle Modules

Monitoring Data

Resources

Concurrent Resources

Connectors

JDBC

JDBC Resources

JDBC Connection Pools

JMS Resources

JNDI

JavaMail Sessions

Resource Adapter Configs

Configurations

imart-config

default-config

JDBC Connection Pools

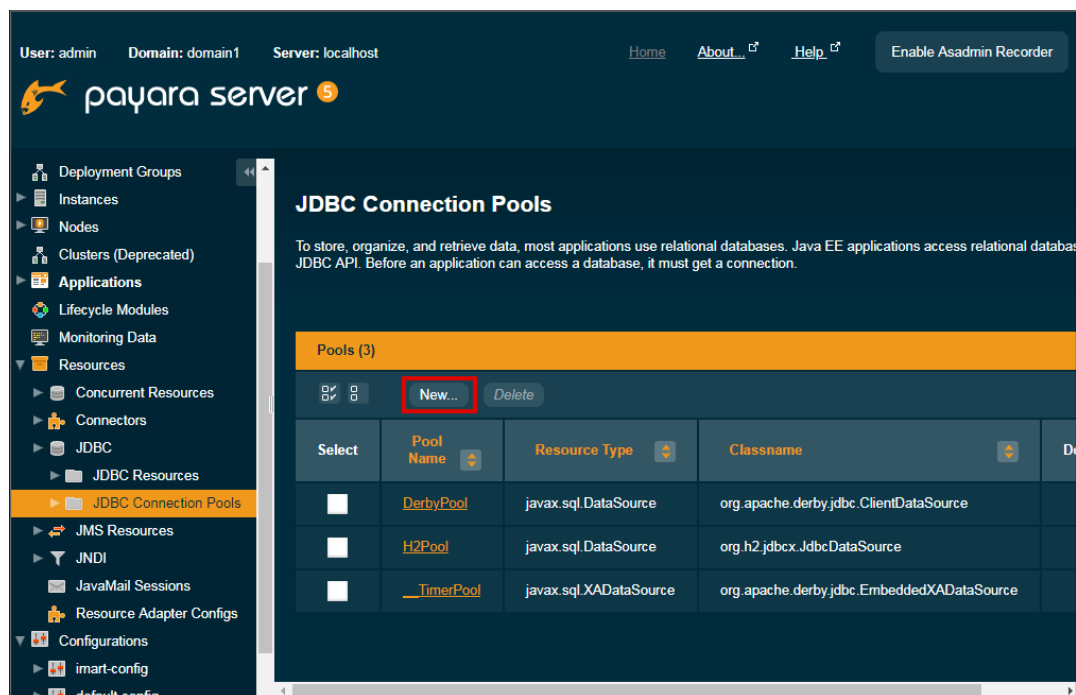
To store, organize, and retrieve data, most applications use relational databases. Java EE applications access relational databases using the JDBC API. Before an application can access a database, it must get a connection.

Pools (3)

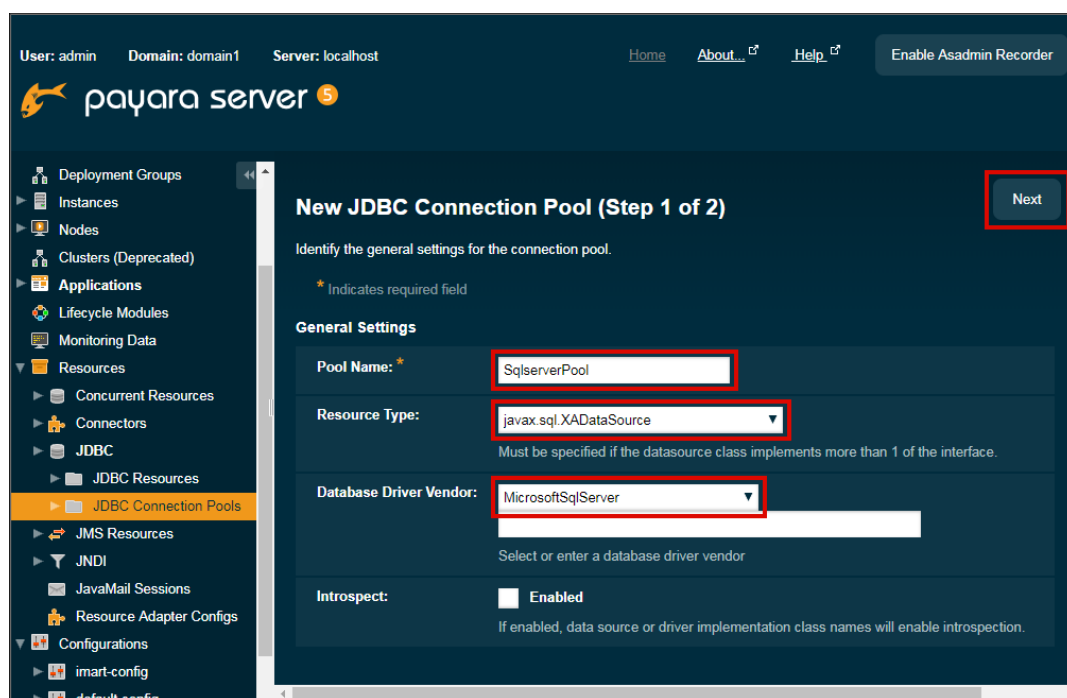
New... Delete

Select	Pool Name	Resource Type	Classname	Description
☐	DerbyPool	javax.sql.DataSource	org.apache.derby.jdbc.ClientDataSource	
☐	H2Pool	javax.sql.DataSource	org.h2.jdbcx.JdbcDataSource	
☐	TimerPool	javax.sql.XADataSource	org.apache.derby.jdbc.EmbeddedXADataSource	

3. [New...] を選択します。



4. [New JDBC Connection Pool (Step 1 of 2)] で以下のように入力して [Next] を選択します。



Pool Name	コネクションプールの名前を入力します。例として、ここでは SqlserverPool と入力します。
Resource Type	リソースタイプを選択します。 javax.sql.XADataSource を選択します。
Database Driver Vendor	JDBC ドライバのベンダを選択します。 MicrosoftSqlServer を選択します。



注意

SQL Server on Linux では分散トランザクションを利用できません。そのため、 javax.sql.XADataSource ではなく javax.sql.DataSource を利用してください。



注意

SQL Server on Linux では分散トランザクションを利用できません。そのため、パーチャルテナント機能は利用できません。



注意

SQL Server (Windows) で分散トランザクション(javax.sql.XADataSource)を利用するには、SQL Server に対して以下の手順を事前に実施する必要があります。

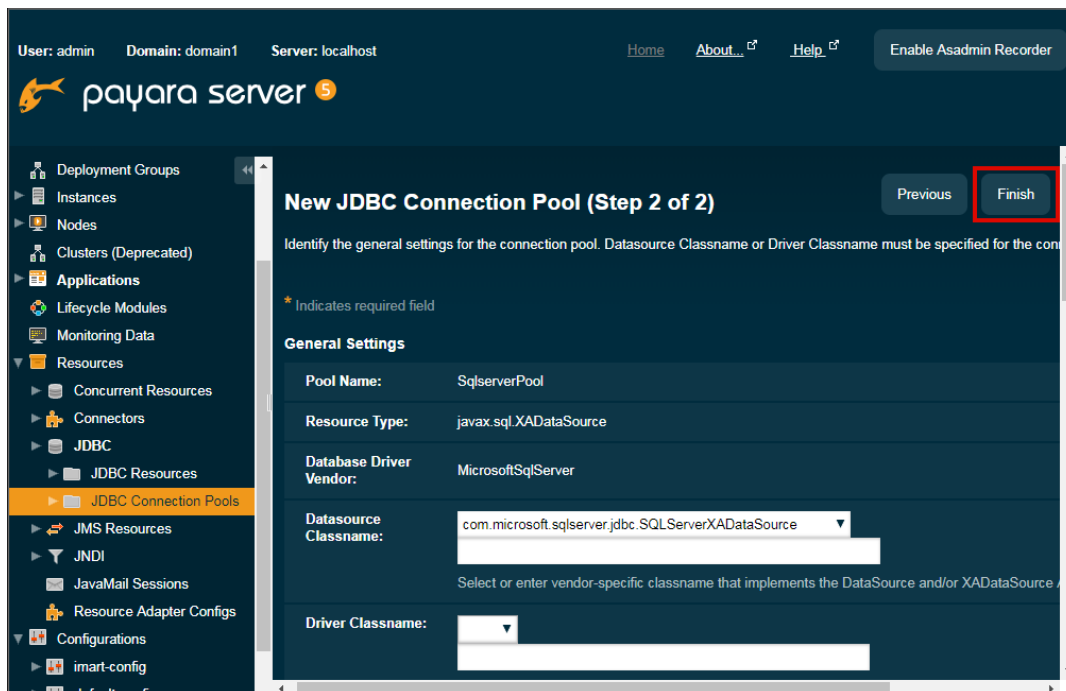
<https://docs.microsoft.com/ja-jp/sql/connect/jdbc/understanding-xa-transactions>

5. [New JDBC Connection Pool (Step 2 of 2)] の [Additional Properties] で以下を変更します。

Select	Name	Value	Description
☐	DatabaseName	iap_db	
☐	User	imart	
☐	URL	jdbc:sqlserver://localhost:1433	
☐	Password	imart	
☐	SelectMethod	cursor	
☐	SendTimeAsDatetime	false	

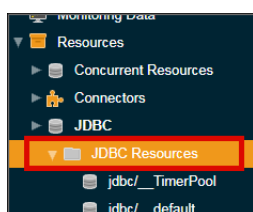
URL	データベースURLを設定します。例として、ここでは <code>jdbc:sqlserver://localhost:1433</code> を設定しています。
User	データベースユーザを設定します。例として、ここでは <code>imart</code> を設定しています。
Password	データベースユーザのパスワードを設定します。例として、ここでは <code>imart</code> を設定しています。
DatabaseName	データベース名を設定します。例として、ここでは <code>iap_db</code> を設定しています。
SelectMethod	参照クエリ実行時にカーソルの利用可否を設定します。 <code>cursor</code> を設定します。
SendTimeAsDatetime	<code>java.sql.Time</code> を <code>datetime</code> 型としてサーバに送信するか否かを設定します。 <code>false</code> を設定します。

6. [New JDBC Connection Pool (Step 2 of 2)] で [Finish] を選択します。

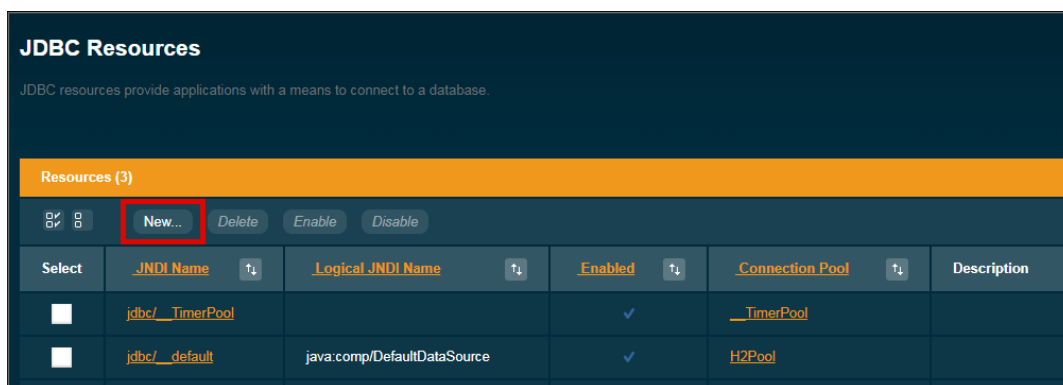


JDBC Resource の作成

1. Payara の管理コンソールにアクセスします。アクセス方法は「[管理コンソールにアクセス](#)」を参照してください。
2. 左ペインのメニューから [Resources] - [JDBC] - [JDBC Resources] を選択します。



3. [New...] を選択します。

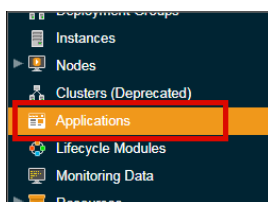


4. [New JDBC Resource] で以下のように入力して [OK] を選択します。

JNDI Name	JNDI 名を設定します。jdbc/default を設定してください。
PoolName	「 Connection Pool の作成 」で作成したコネクションプールを設定します。
Targets	「 インスタンスの追加 」で作成したインスタンスを [Selected Targets] に設定します。

war ファイルのデプロイ

1. Payara の管理コンソールにアクセスします。アクセス方法は「[管理コンソールにアクセス](#)」を参照してください。
2. 左ペインのメニューから [Applications] を選択します。



3. [Deploy...] を選択します。

4. [Location] に war ファイルを選択します。

Deploy Applications or Modules OK Cancel

Specify the location of the application or module to deploy. An application can be in a packaged file or specified as a directory.

* Indicates required field

Location:

☒ Packaged File to Be Uploaded to the Server

ファイルを選択 imart.war

☐ Local Packaged File or Directory That Is Accessible from Payara Server

Browse Files... Browse Folders...

5. 以下のように入力して [OK] を選択します。

Deploy Applications or Modules OK Cancel

Specify the location of the application or module to deploy. An application can be in a packaged file or specified as a directory.

* Indicates required field

Location:

☒ Packaged File to Be Uploaded to the Server

ファイルを選択 imart.war

☐ Local Packaged File or Directory That Is Accessible from Payara Server

Browse Files... Browse Folders...

Type: Web Application

Context Root: imart
Path relative to server's base URL.

Application Name: imart

Status: ☒ Enabled
Allows users to access the application.

Implicit CDI: ☒ Enabled
Implicit discovery of CDI beans

CDI Development Mode: ☐ Enabled
Allows to inspect the application CDI components at runtime

Availability: ☒ Enabled
Controls whether availability is enabled for web sessions and for stateful session bean (SFSB) checkpointing and potentially passivation.

Precompile JSPs: ☒
Precompiles JSP pages during deployment.

Run Verifier: ☐
Verifies the syntax and semantics of the deployment descriptor. Verifier packages must be installed.

Force Redeploy: ☐
Forces redeployment even if this application has already been deployed or already exists.

Keep State: ☐
Retains web sessions, SFSB instances, and persistently created EJB timers between redeployments.

Deployment Order:
A number that determines the loading order of the application at server startup. Lower numbers are loaded first. The default is 100.

Libraries:
A comma-separated list of library JAR files. Specify the library JAR files by their relative or absolute paths. Specify relative paths relative to `instance-root/lib/app/libs`. The libraries are made available to the application in the order specified.

Description:

Targets

Available Targets:

server

Add > Add All >> < Remove << Remove All

Selected Targets:

imart-instance

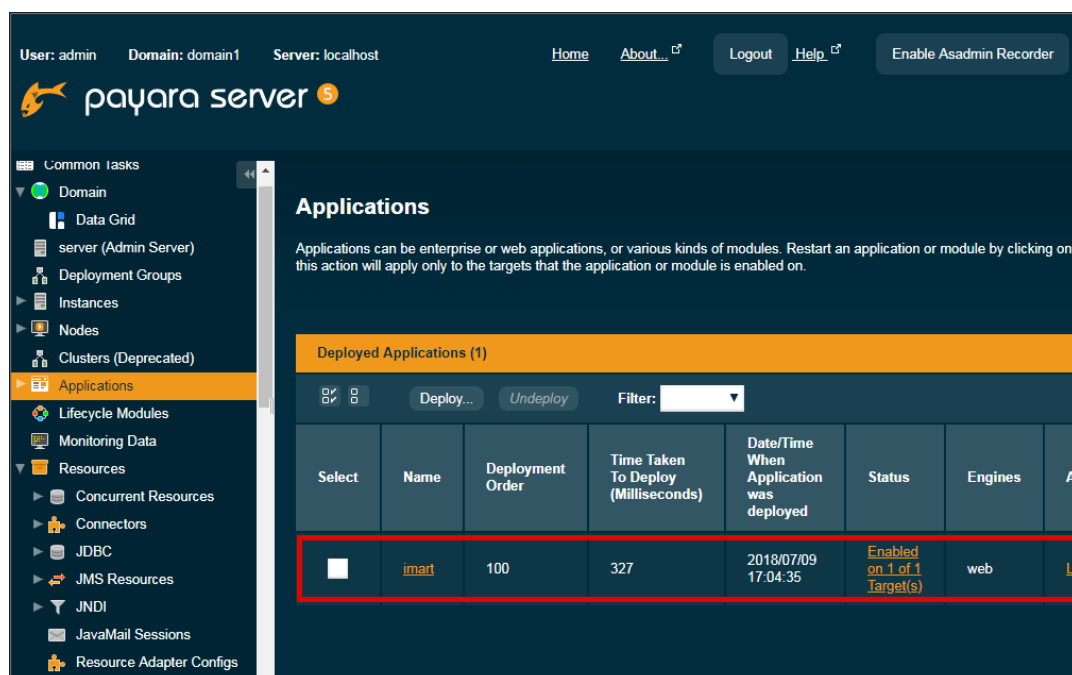
Availability

[Enabled] を ON にします。

Precompile JSPs ON にします。

Targets 「[インスタンスの追加](#)」で作成したインスタンスを [Selected Targets] に設定します。

6. war がデプロイされました。



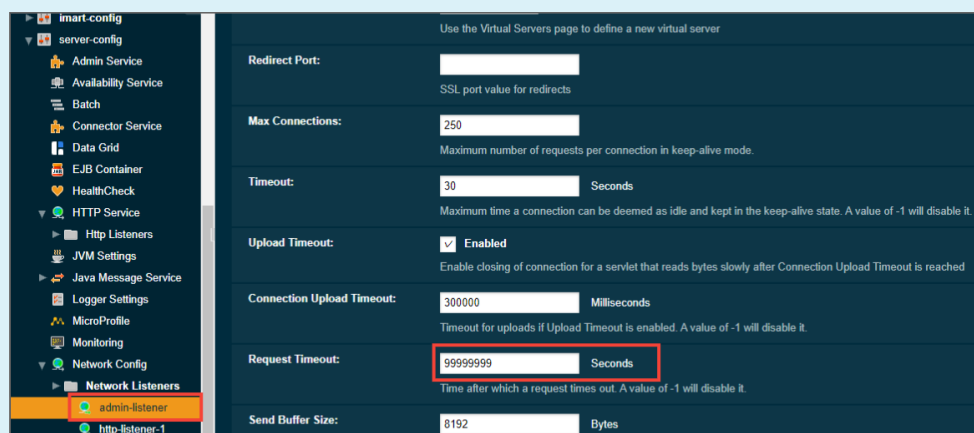
コラム

デプロイ完了後に %PAYARA_HOME%/glassfish/domains/domain1/logs/server.log に以下のログが出力されます。

```
| [2023-05-15T13:26:15.520+0900] [Payara 5.2022.5] [情報] [] [javax.enterprise.system.core] [tid:
_ThreadID=203 _ThreadName=admin-thread-pool::admin-listener(3)] [timeMillis: 1684124775520] [levelValue:
800] []
| imart was successfully deployed in 55,448 milliseconds.]]
```

コラム

デプロイが失敗した場合には、管理コンソールの Request Timeout に大きな値を設定することで回避できる場合があります。Request Timeout は、左ペインのメニューから [Configurations] - [server-config] - [Network Config] - [Network Listeners] - [admin-listener] を選択し、HTTP タブで変更できます。



インスタンスの起動と停止

項目

- インスタンスの起動
- インスタンスの停止
- インスタンスの起動確認
- インスタンスのログ

インスタンスの起動と停止方法について記述します。

i コラム

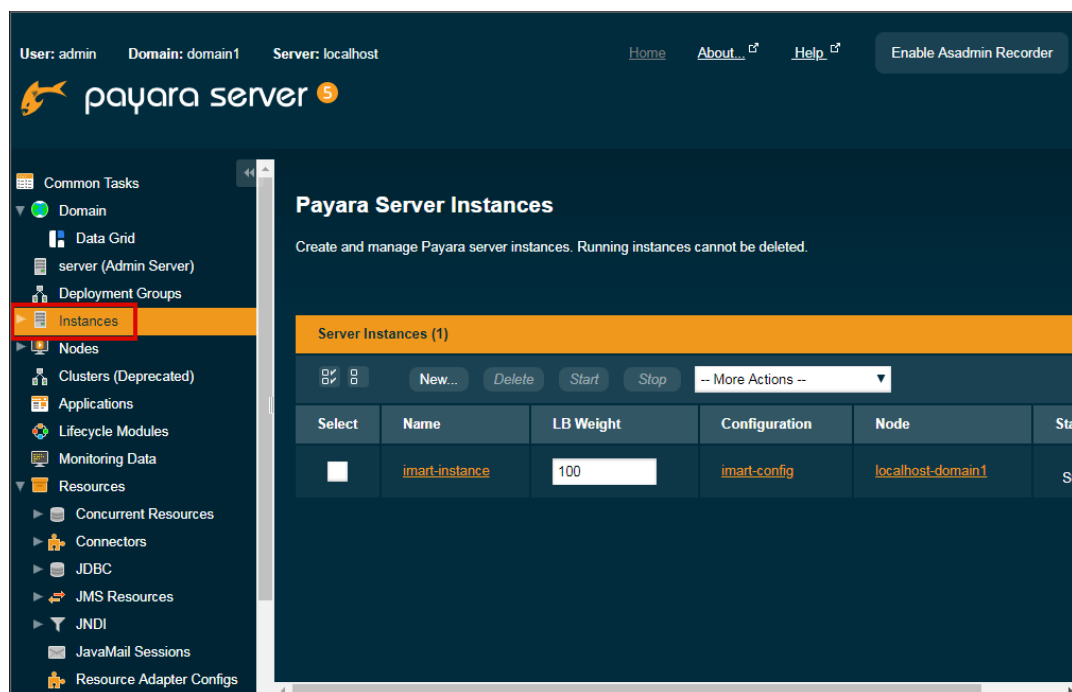
インスタンスの停止と起動によるアプリケーションの再起動を行った場合、以下の問題により非同期スレッドが実行されない可能性があります。

その場合、Payara Server 自体を再起動してください。

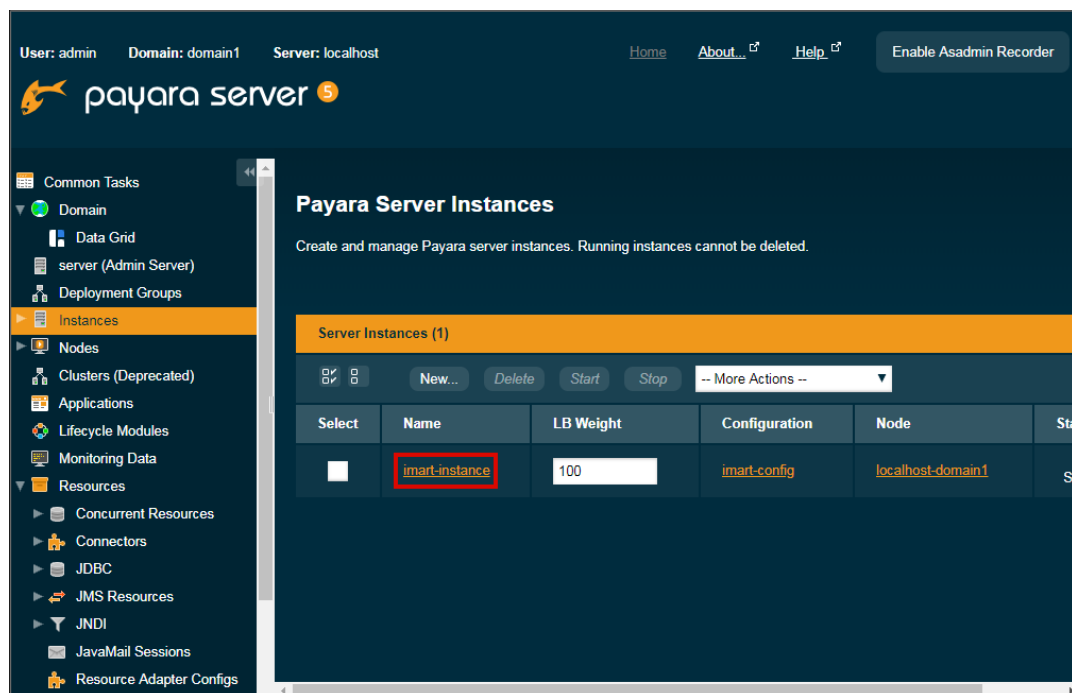
<https://github.com/payara/Payara/pull/2453>

インスタンスの起動

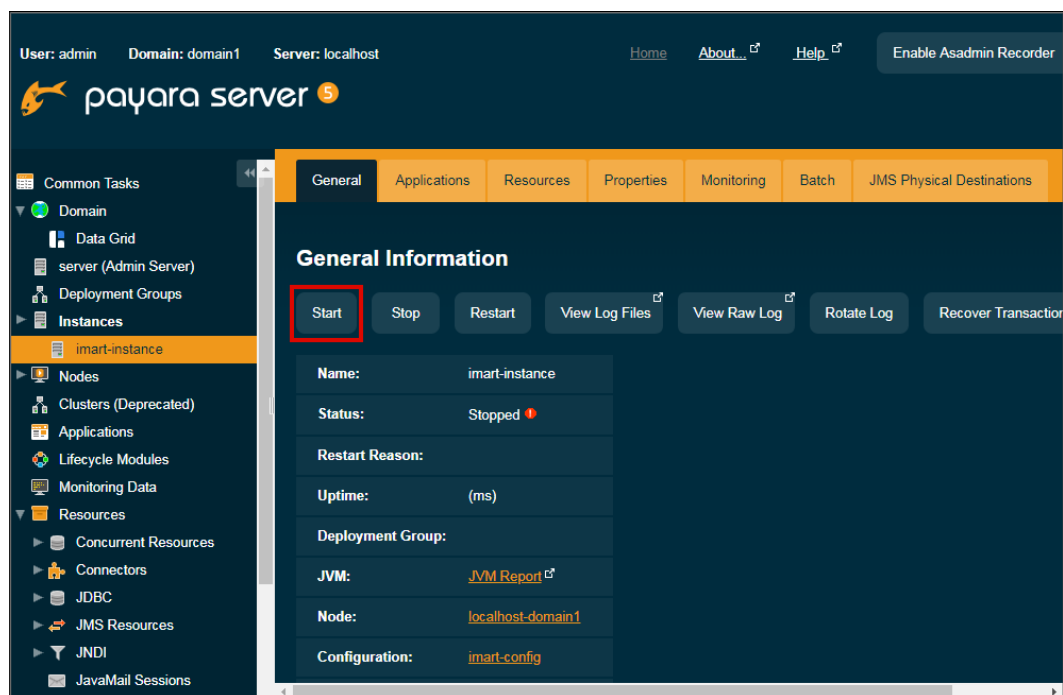
1. Payara の管理コンソールにアクセスします。アクセス方法は「[管理コンソールにアクセス](#)」を参照してください。
2. 左ペインのメニューから [Instances] を選択します。



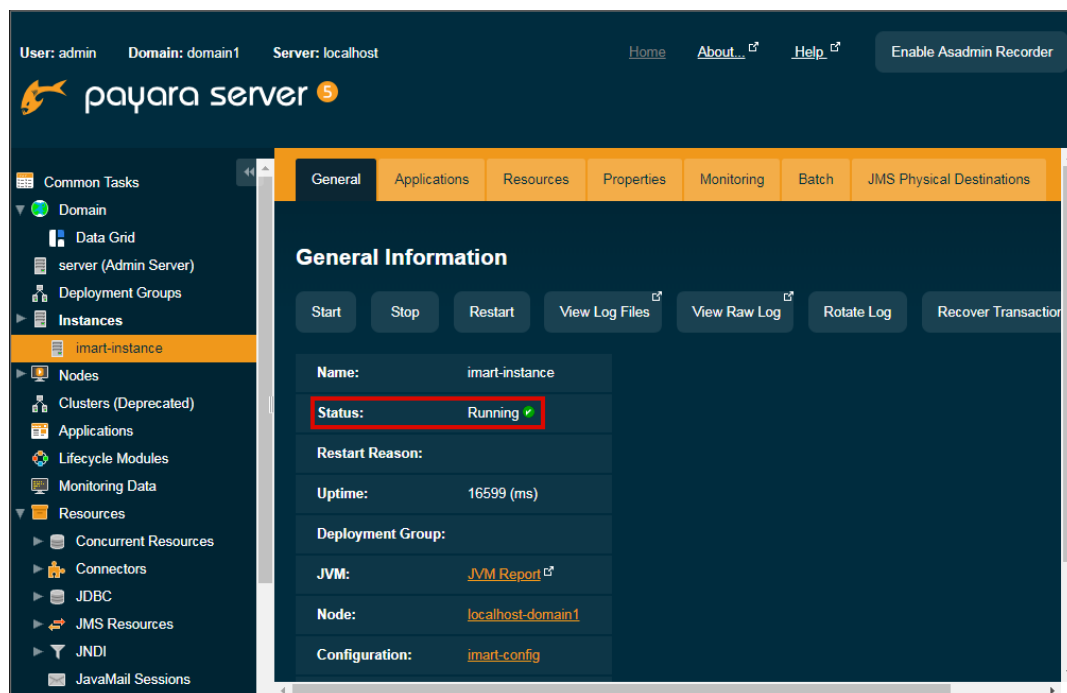
3. 「[インスタンスの追加](#)」で追加したインスタンスを選択します。例として、ここでは [imart-instance] を選択します。



4. [Start] を選択します。

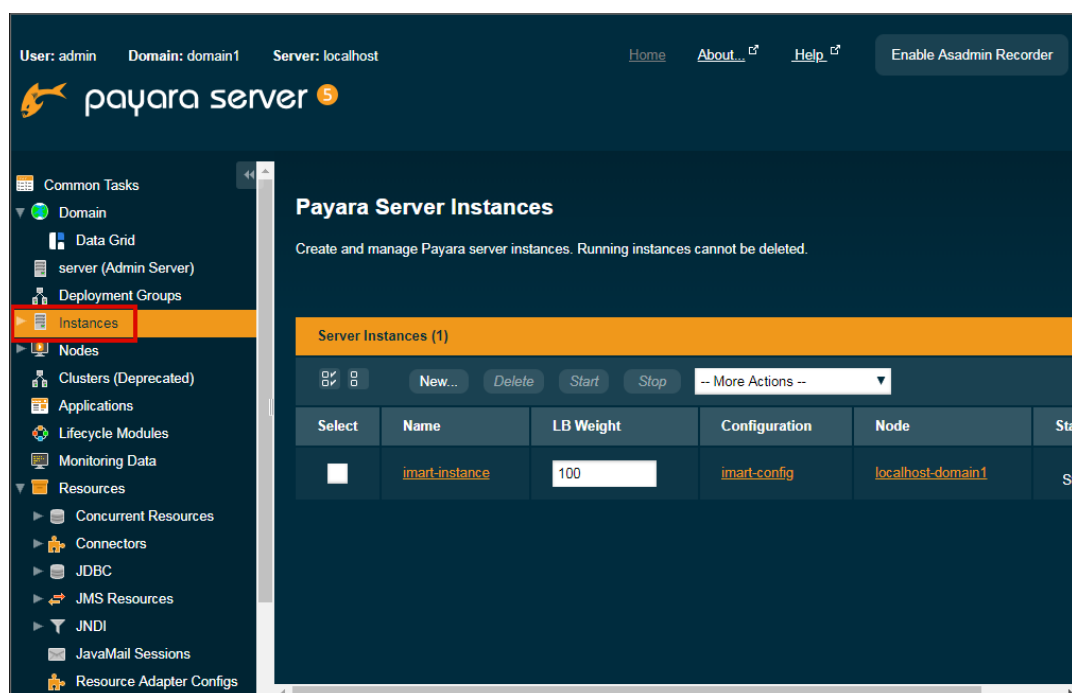


5. 起動が完了すると Status に Running と表示されます。

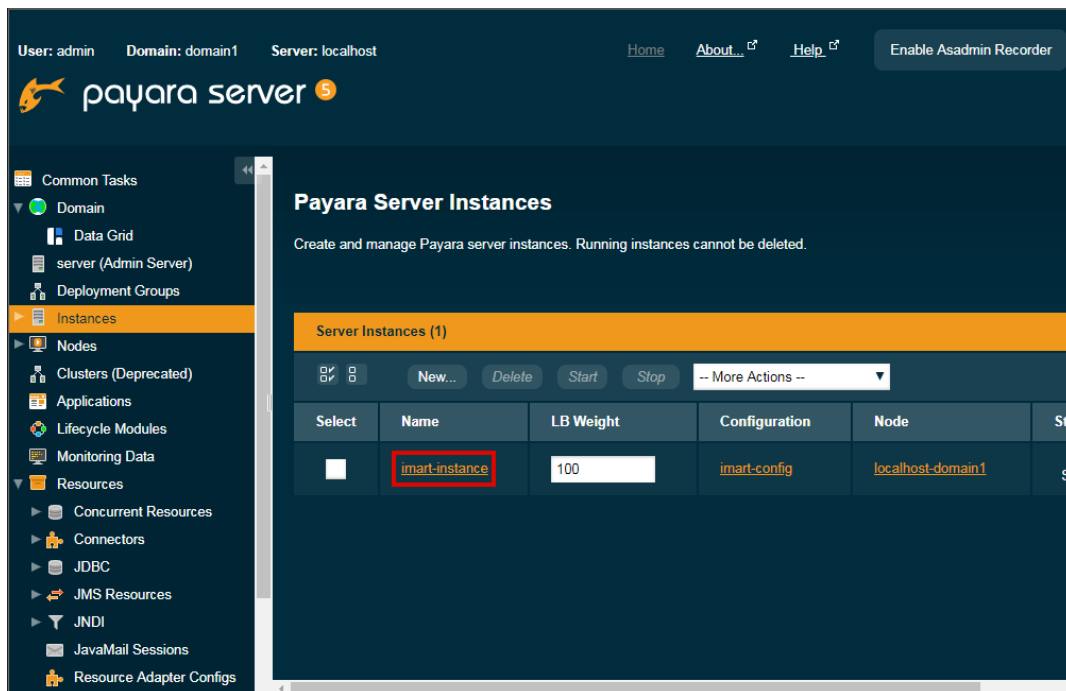


インスタンスの停止

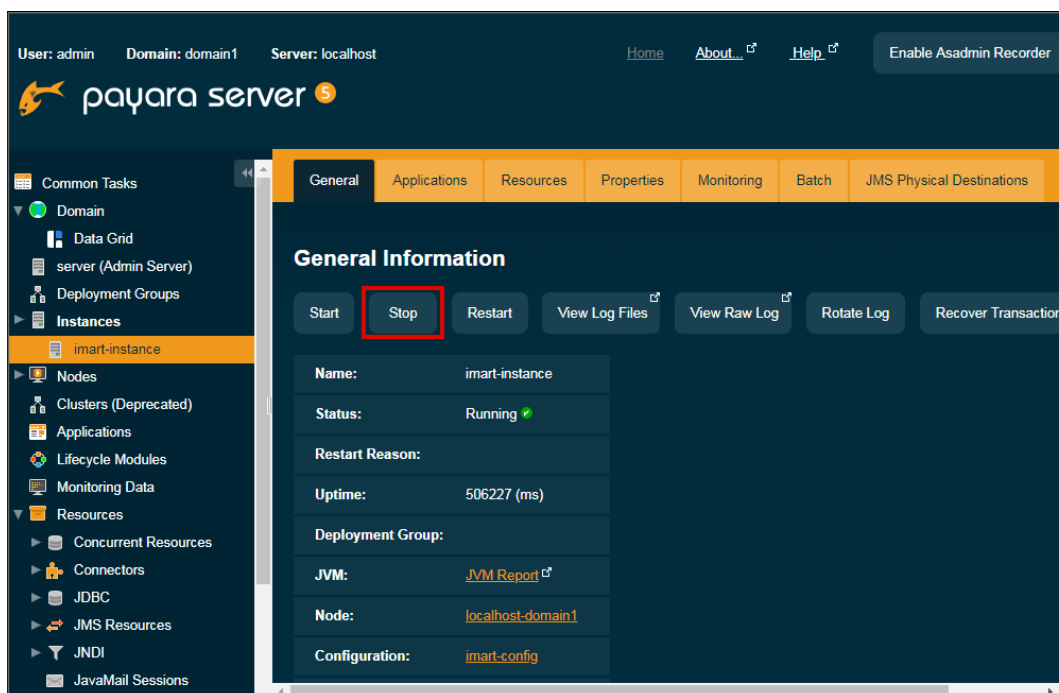
1. Payara の管理コンソールにアクセスします。アクセス方法は「[管理コンソールにアクセス](#)」を参照してください。
2. 左ペインのメニューから [Instances] を選択します。



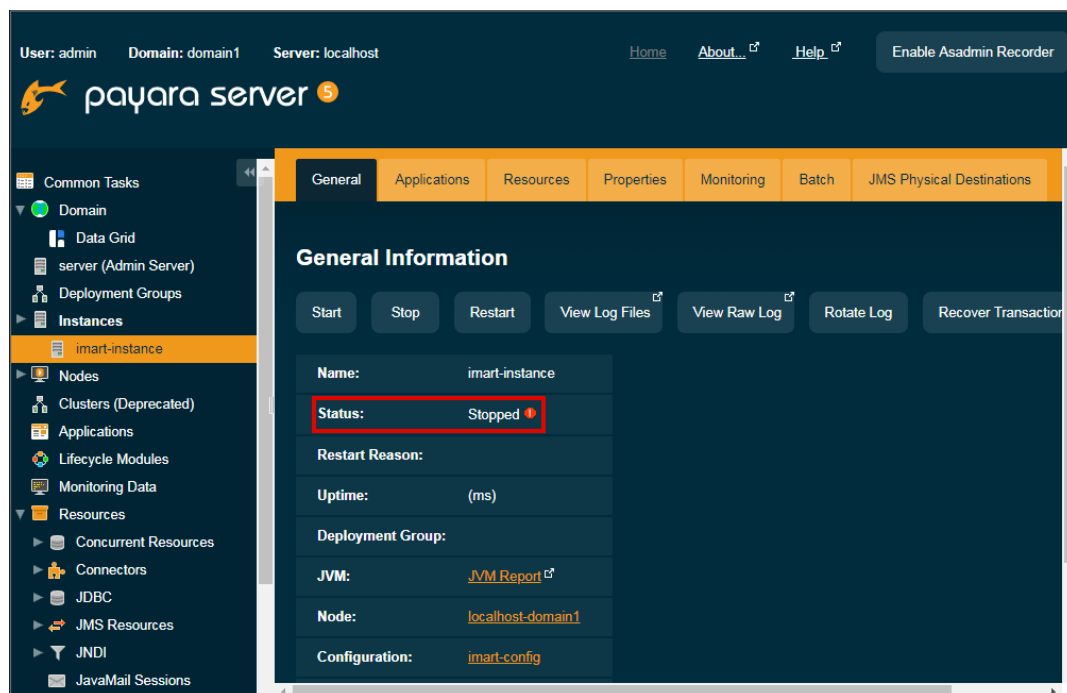
3. 「[インスタンスの追加](#)」で追加したインスタンスを選択します。例として、ここでは [imart-instance] を選択します。



4. [Stop] を選択します。

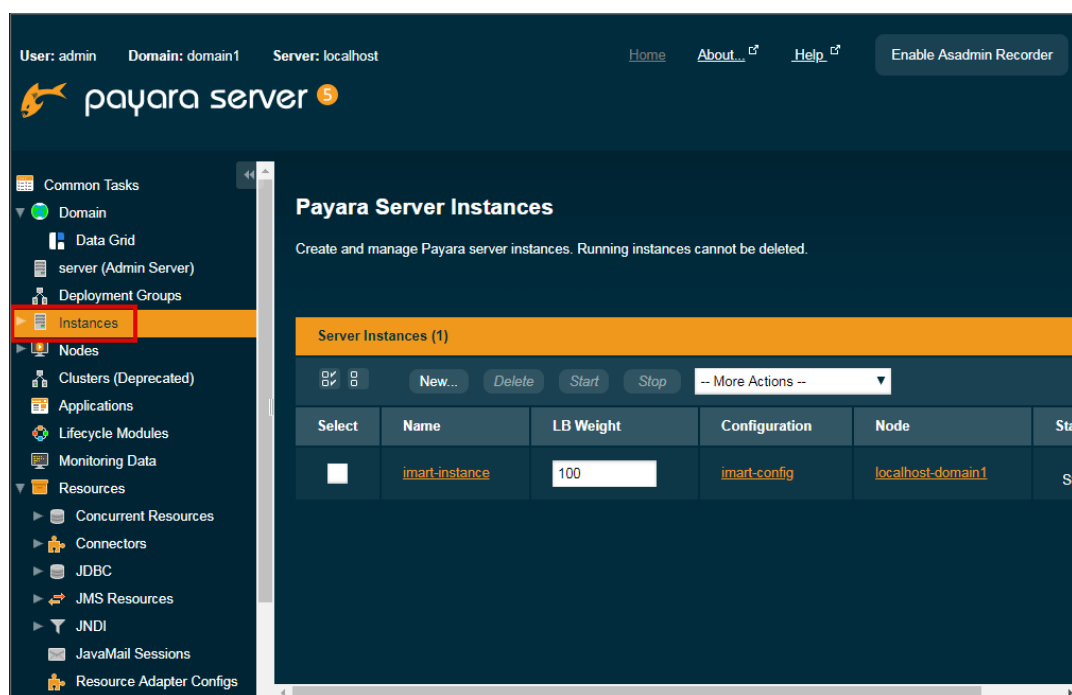


5. 停止が完了すると Status に Stopped と表示されます。

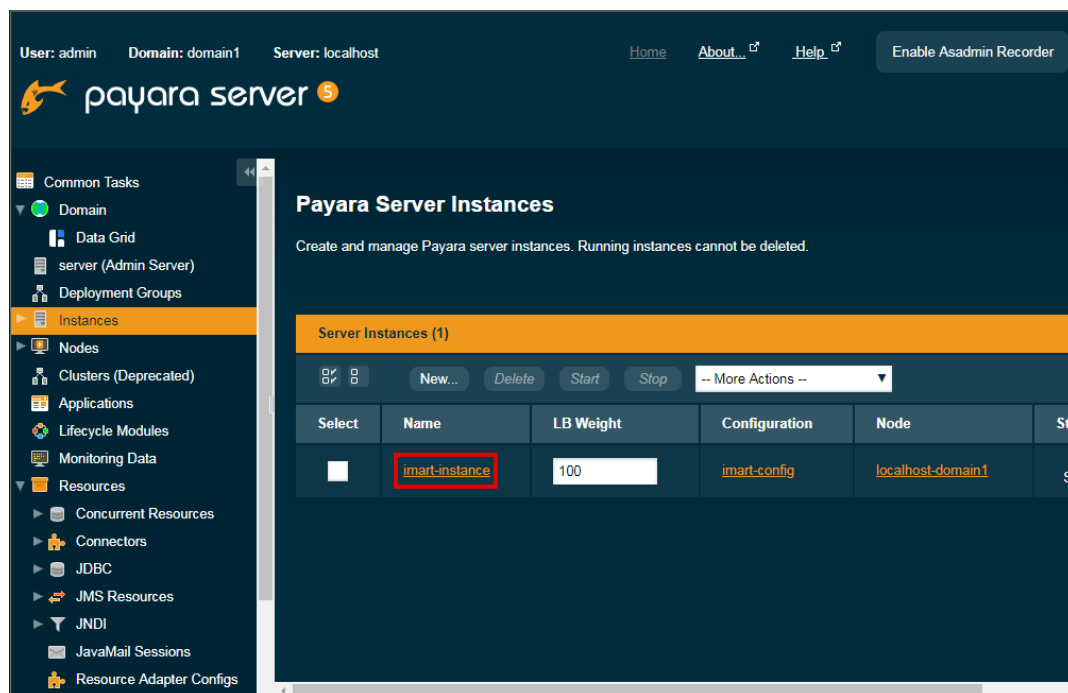


インスタンスの起動確認

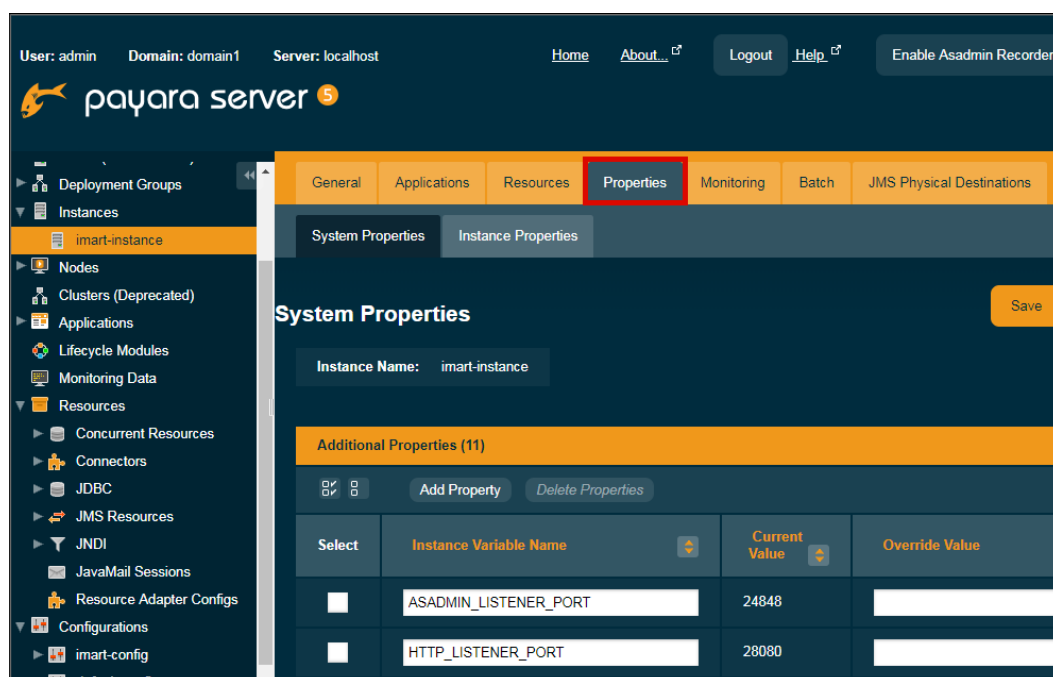
1. 左ペインのメニューから [Instances] を選択します。



2. [imart-instance] を選択します。



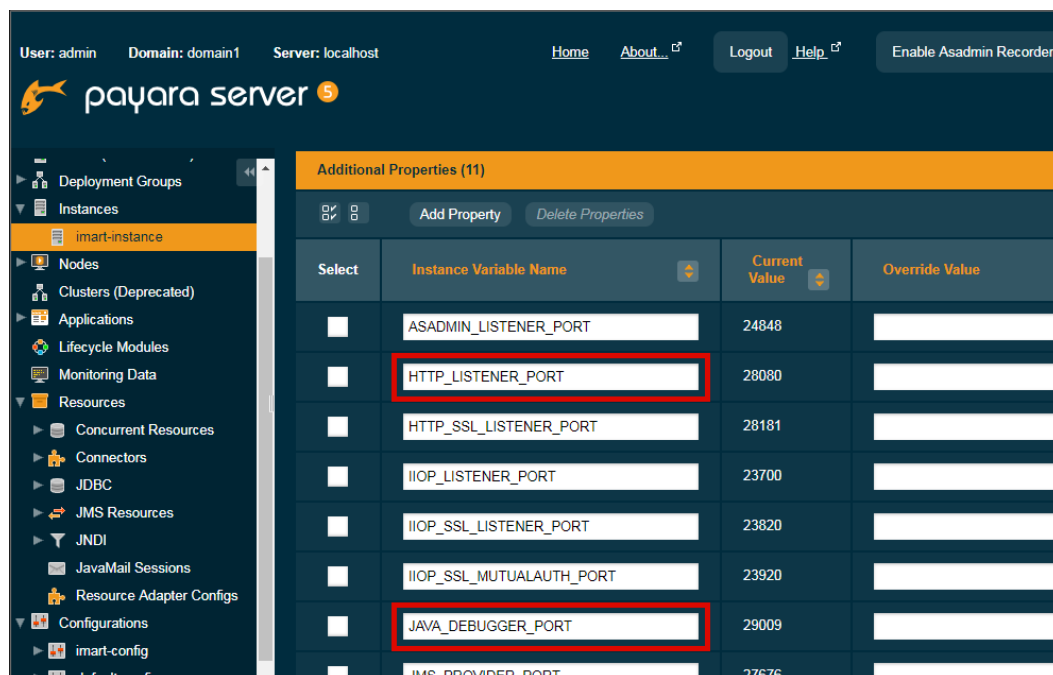
3. [Properties] を選択します。



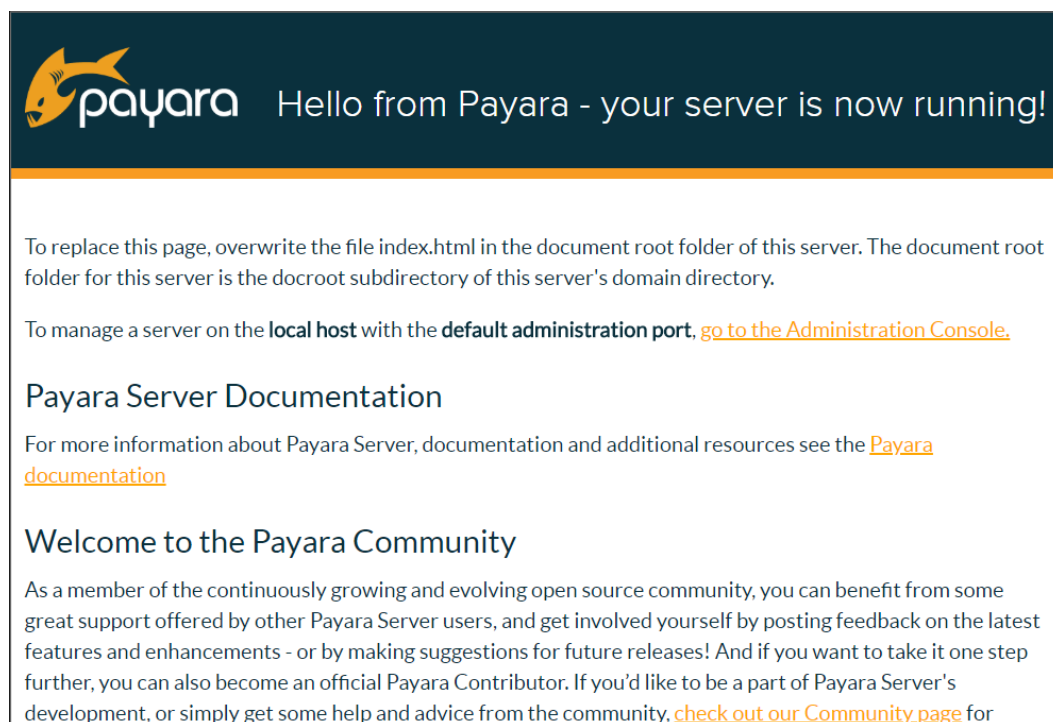
4. 以下のプロパティを確認します。

HTTP_LISTENER_PORT imart-instance が起動するポートです。

JAVA_DEBUGGER_PORT imart-instance の Java リモートデバッグのポートです。リモートデバッグを利用する場合はこのポートに接続してください。



5. localhost:28080 ポートにアクセスします。
 以下のような画面が表示されることを確認してください。



コラム

コンテキストルート名 [imart] で war をデプロイした場合、以下の URL からログインできます。

- <http://localhost:28080/imart/login> (一般ユーザのログイン画面)
- <http://localhost:28080/imart/system/login> (システム管理者のログイン画面)



コラム

localhost の部分は Payara を実行しているサーバの IP アドレスに置き換えてください。

インスタンスのログ

以下のファイルに出力されます。

%PAYARA_HOME%/glassfish/nodes/localhost-domain1/imart-instance/logs/server.log

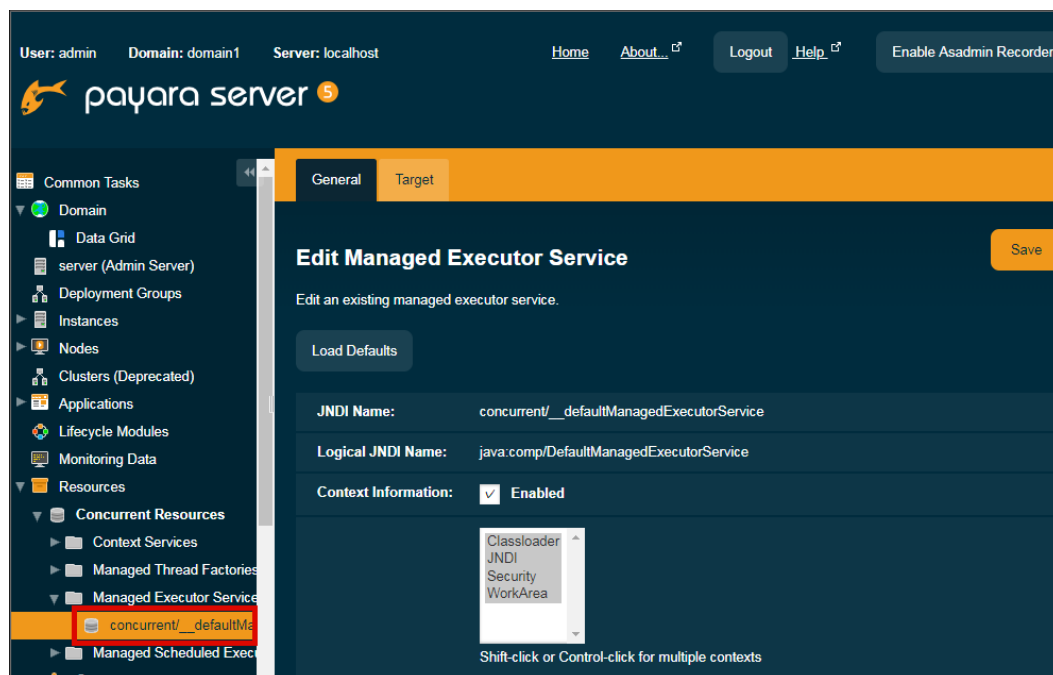
Payara Server では非同期処理のタスク実行エンジンとして Concurrency Utilities for Java EE を利用しています。非同期処理機能では JNDI を通じて Concurrency Utilities の ExecutorService を取得しています。本章では ExecutorService の設定を行います。

項目

- Executor Service の設定

Executor Service の設定

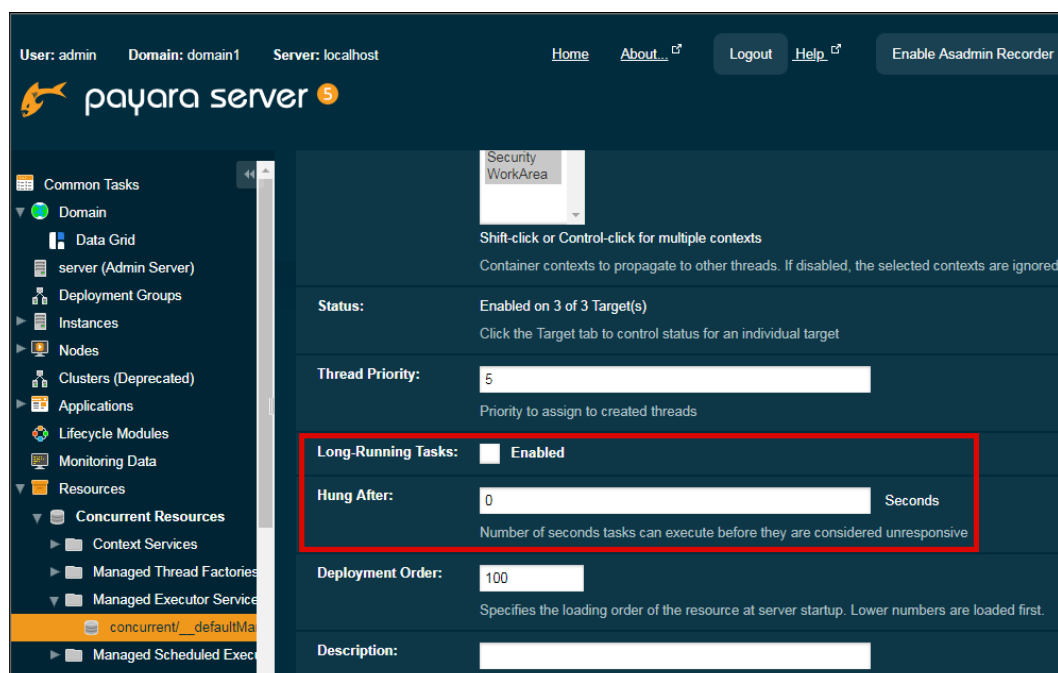
1. 左ペインのメニューから [Resources] - [Concurrent Resources] - [Managed Executor Services] - [concurrent/_defaultManagedExecutorService] を選択します。



2. 以下の設定を確認します。

Long-Running Tasks Hung After に 1 以上の値を設定する場合、選択して有効化してください。
Hung After に 0 を設定する場合、どちらでも構いません。

Hung After 0 を指定してください。





注意

Hung After に 1 以上の値を設定し、かつ Long-Running Tasks を無効化している場合、非同期スレッドが Hung After 秒以上実行後に以下のような警告が表示されます。

```
[2018-07-04T17:51:24.433+0900] [Payara 5.182] [警告] [AS-CONCURRENT-00001] [javax.enterprise.concurrent]
[tid: _ThreadID=327 _ThreadName=glassfish-internal-managedThreadFactory-Thread-1] [timeMillis:
1530694284433] [levelValue: 900] [[
Task [MyTask@61a84554] has been running on thread [concurrent/_defaultManagedExecutorService-
managedThreadFactory-Thread-1] for 52 seconds, which is more than the configured hung task threshold of 5
seconds in [concurrent/_defaultManagedExecutorService.]]]
```

intra-mart Accel Platform のテナント環境を構築します。

具体的な手順は「[テナント環境セットアップ](#)」を参照してください。

付録

Payara のサービス化

Payara のサービス化手順について説明します。

項目

- サービスの登録
- サービスの削除
- サービスの起動
- サービスの終了

サービスの登録

1. create-service domain1 コマンドを実行します。

- Linux

```
$PAYARA_HOME/bin/asadmin create-service domain1
```

- Windows

```
%PAYARA_HOME%\bin\asadmin.bat create-service domain1
```



コラム

管理者権限で実行してください。



コラム

java へのパスが通っていない場合、以下の手順を参考にして、<%PAYARA_HOME%/glassfish/config/asenv.conf> ファイルに AS_JAVA の設定を追加します。

参考：[SSH ノードとなるサーバに Payara をインストールする](#)

サービスの削除

- Linux

```
rm -f /etc/rc.d/init.d/payara_domain1
rm -f /etc/rc.d/rc0.d/K20payara_domain1
rm -f /etc/rc.d/rc1.d/K20payara_domain1
rm -f /etc/rc.d/rc2.d/S20payara_domain1
rm -f /etc/rc.d/rc3.d/S20payara_domain1
rm -f /etc/rc.d/rc4.d/S20payara_domain1
rm -f /etc/rc.d/rc5.d/S20payara_domain1
rm -f /etc/rc.d/rc6.d/K20payara_domain1
```

- Windows

```
sc delete domain1
```



コラム

管理者権限で実行してください。

サービスの起動

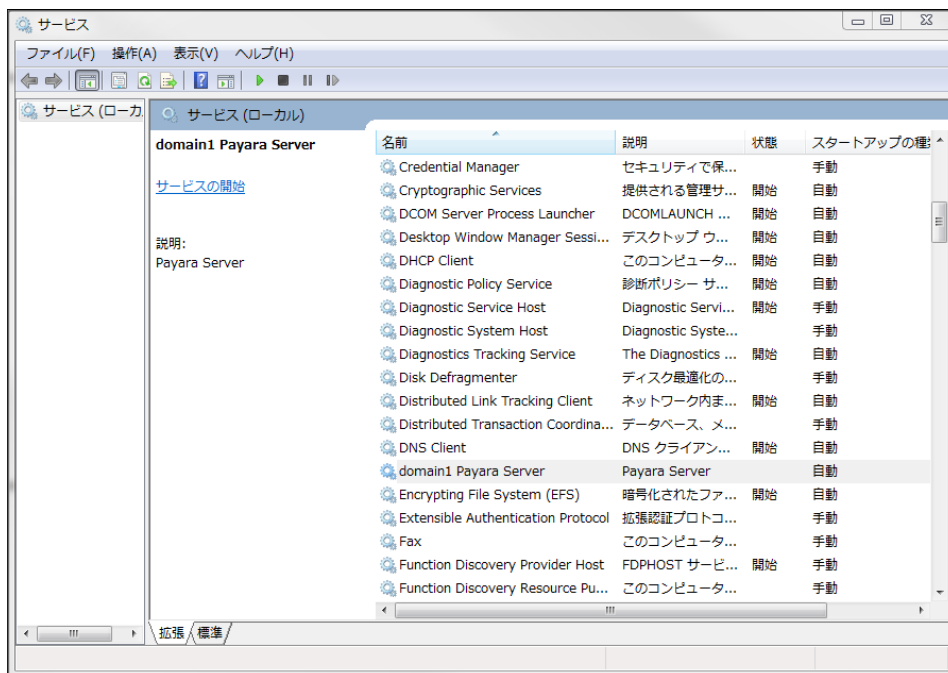
Linux

以下のコマンドを実行してください。

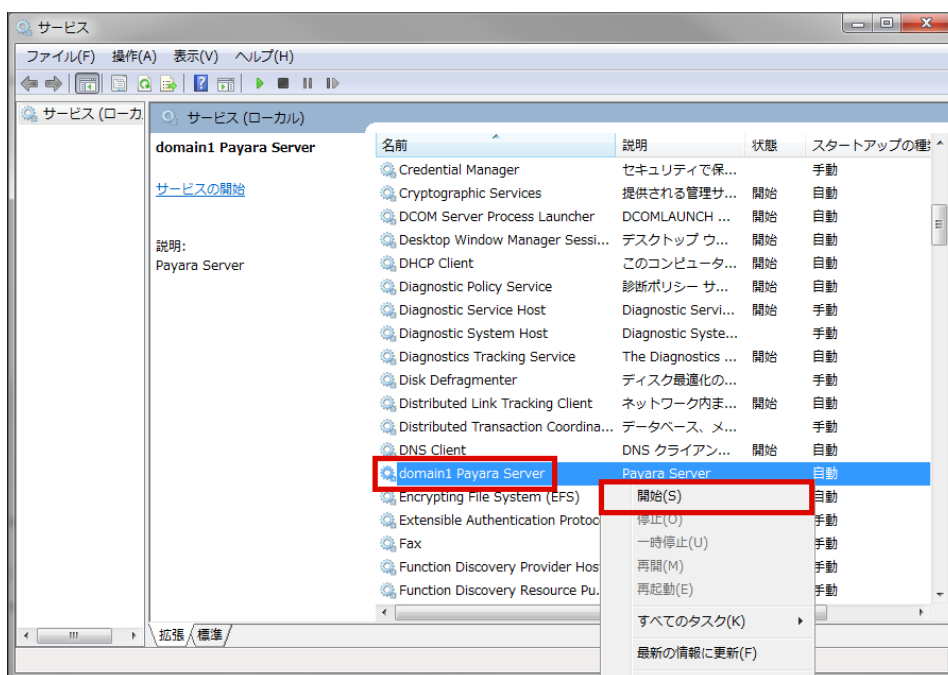
```
service payara_domain1 start
```

Windows

1. サービス管理ツール(services.msc)を起動します。



2. [domain1 Payara Server] を右クリックし、[開始] をクリックします。



サービスの終了

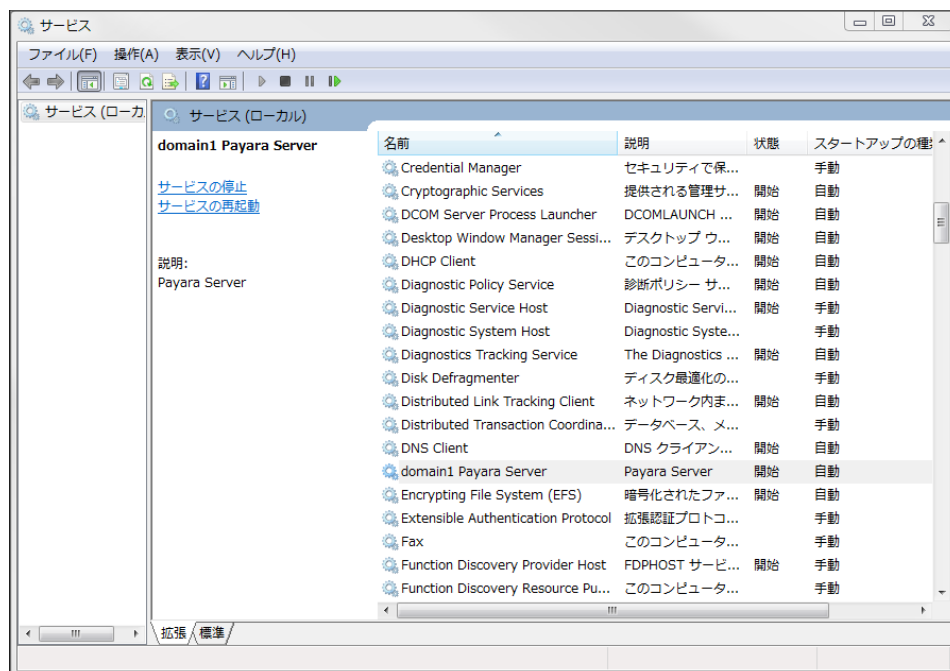
Linux

以下のコマンドを実行してください。

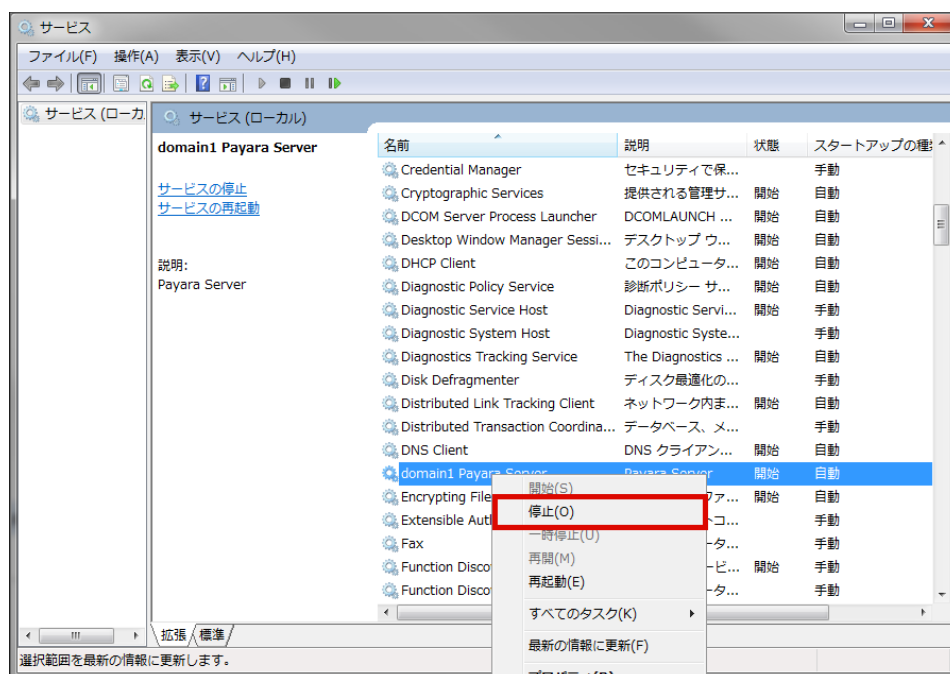
```
service payara_domain1 stop
```

Windows

1. サービス管理ツール(services.msc)を起動します。



2. [domain1 Payara Server] を右クリックし、[停止] をクリックします。



NIC が複数ある場合

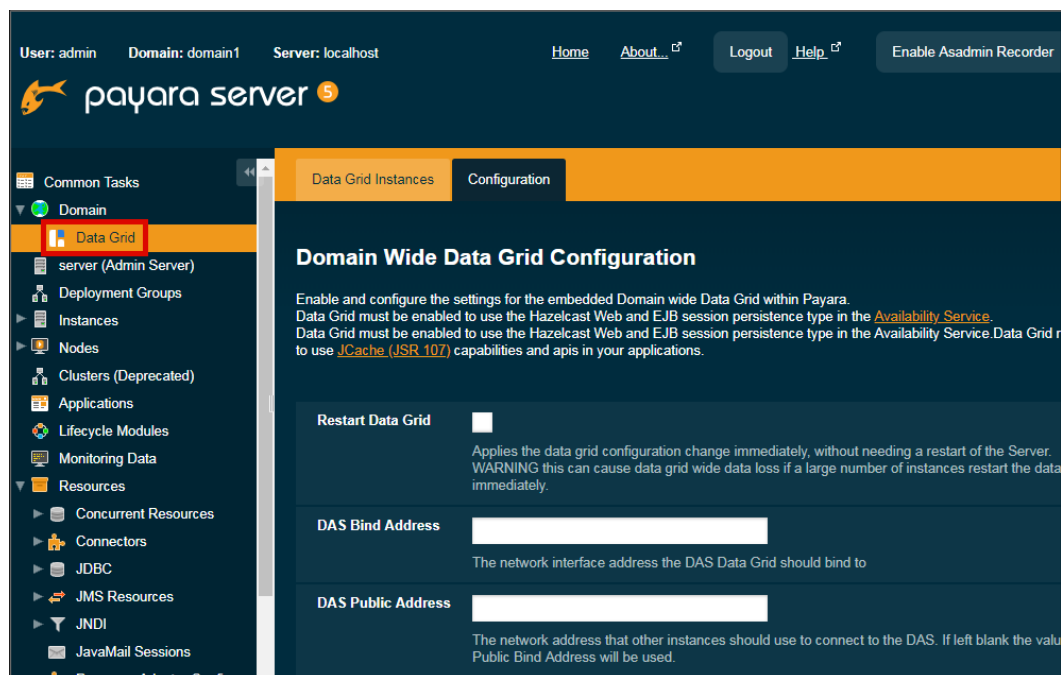
NIC が複数存在する環境の設定について説明します。

項目

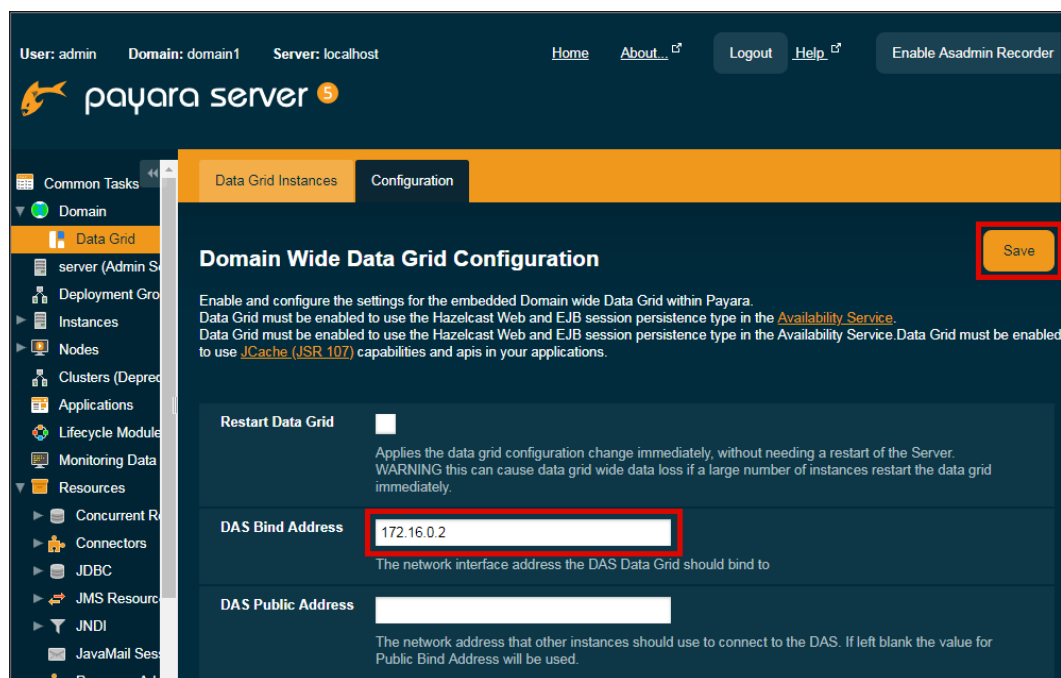
- DAS Bind Address を設定する
- jgroups.bind_addr を設定する

DAS Bind Address を設定する

1. 左ペインのメニューから [Domain] - [Data Grid] を選択します。

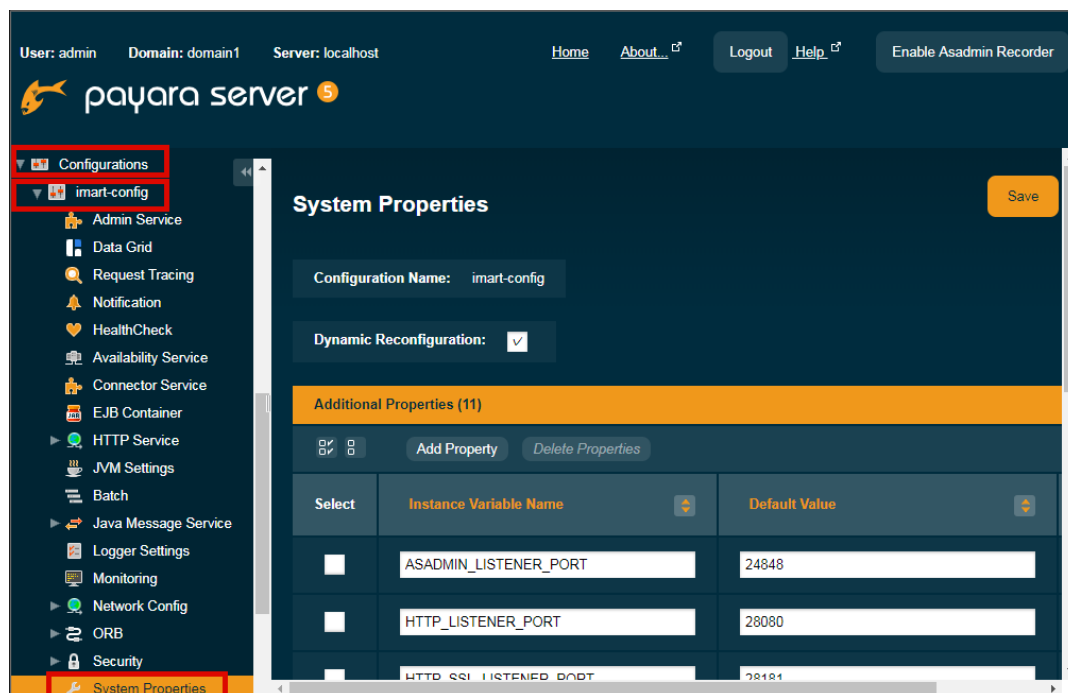


2. [DAS Bind Address] に使用する NIC の IP アドレスを設定し、[Save] をクリックします。

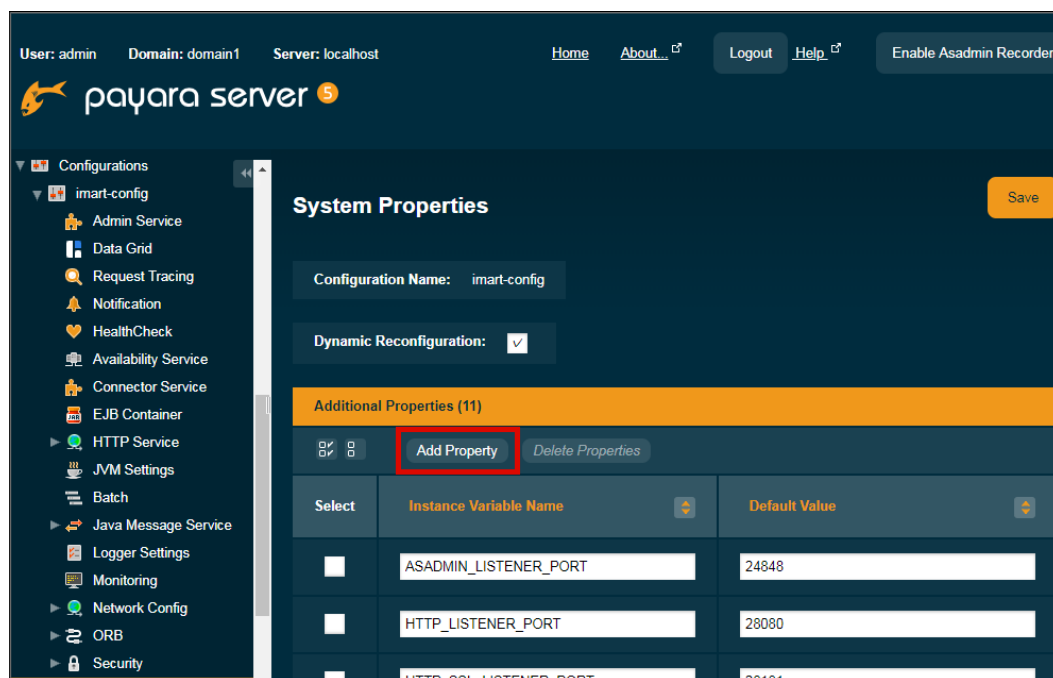


jgroups.bind_addr を設定する

1. 左ペインのメニューから [Configurations] - [imart-config] を選択し、[System Properties] をクリックします。



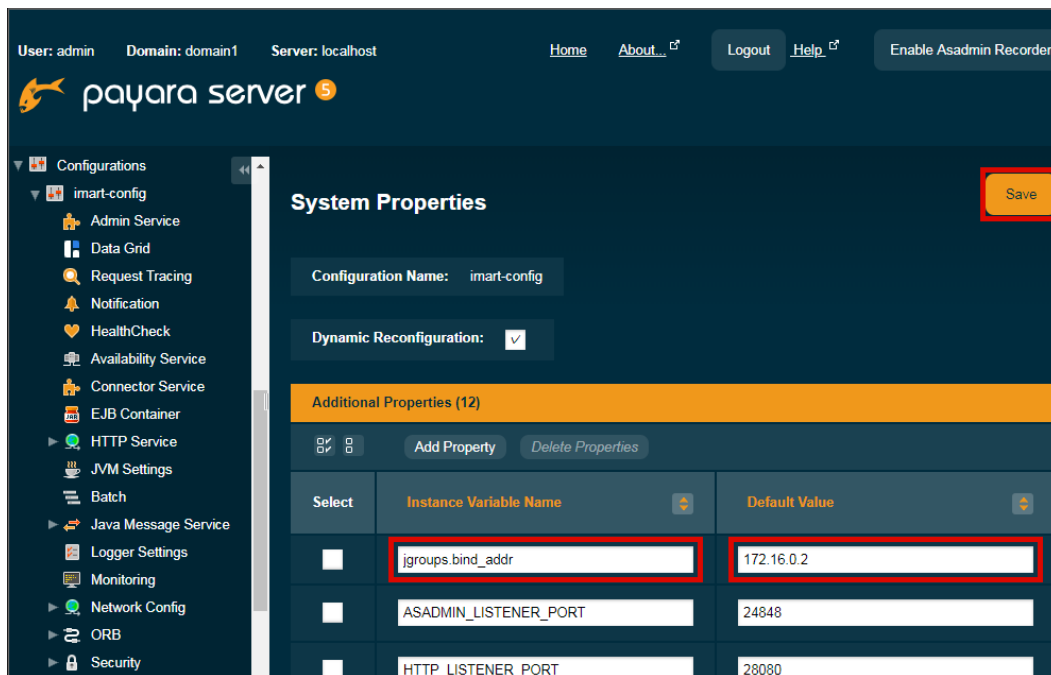
2. [Add Property] をクリックします。



3. 下記のように入力し、[Save] をクリックします。

Instance Variable Name jgroups.bind_addr

Default Value 使用する NIC の IP アドレスを設定します。
例として、ここでは 172.16.0.2 と入力します。



Payara のクラスタリング

Payara のクラスタリング環境の構築手順について説明します。

項目

- [DAS ノードとなるサーバに Payara をインストールする](#)
- [SSH ノードとなるサーバに Payara をインストールする](#)
- [DAS ノードの Payara を起動する](#)
- [DAS ノードの Payara の管理コンソールにアクセスする](#)
- [SSH ノードの設定を行う](#)
- [SSH ノードインスタンスを作成する](#)
- [Deployment Group を作成する](#)
- [Deployment Group を終了する](#)
- [Deployment Group を起動する](#)
- [Domain Data Grid を確認する](#)
- [デプロイメントグループを JDBC リソースの対象に設定する](#)
- [アプリケーションをデプロイする](#)
- [インスタンスの起動確認](#)

本章では以下の 2 台構成でセットアップを行う手順について説明します。

	マシン1	マシン2
プライベートIPアドレス	172.16.0.2	172.16.0.3
構成	DAS + インスタンス1	インスタンス2
名称	DAS ノード	SSH ノード

本章では、マシン1を「DAS ノード」、マシン2「SSH ノード」を呼びます。

DAS ノードとなるサーバに Payara をインストールする

1. DAS ノードとなるサーバに以下のセットアップを実施します。
 - [Payara のインストール](#)
 - [Payara の設定](#)



コラム

DAS ノードとなるサーバにはスタンドアローンと同様のセットアップを実施します。

SSH ノードとなるサーバに Payara をインストールする

1. SSH ノードとなるサーバに以下のセットアップを実施します。

- [Payara をインストールする](#)



コラム

SSH ノードとなるサーバには Payara のインストールと JDBC ドライバの配置を実施します。



コラム

SSH ノード上の Payara の起動や停止は DAS ノードとなるサーバから実施します。そのため、このステップでは Payara と JDBC ドライバの配置のみを行ってください。

2. <%PAYARA_HOME%/glassfish/config/asenv.conf> ファイルに AS_JAVA の設定を追加します。

```
...
AS_IMQ_LIB="../../mq/lib"
AS_IMQ_BIN="../../mq/bin"
AS_CONFIG="../../config"
AS_INSTALL=".."
AS_DEF_DOMAINS_PATH="../../domains"
AS_DEF_NODES_PATH="../../nodes"
AS_DERBY_INSTALL="../../javadb"
AS_H2_INSTALL="../../h2db"
AS_JAVA="/usr/local/jdk"
```



コラム

AS_JAVA に指定する Java のパスには、\$AS_JAVA/bin/java ファイルが存在するようなパスを設定してください。
例として、ここでは /usr/local/jdk と設定します。
この設定により、/usr/local/jdk/bin/java が利用されます。

DAS ノードの Payara を起動する

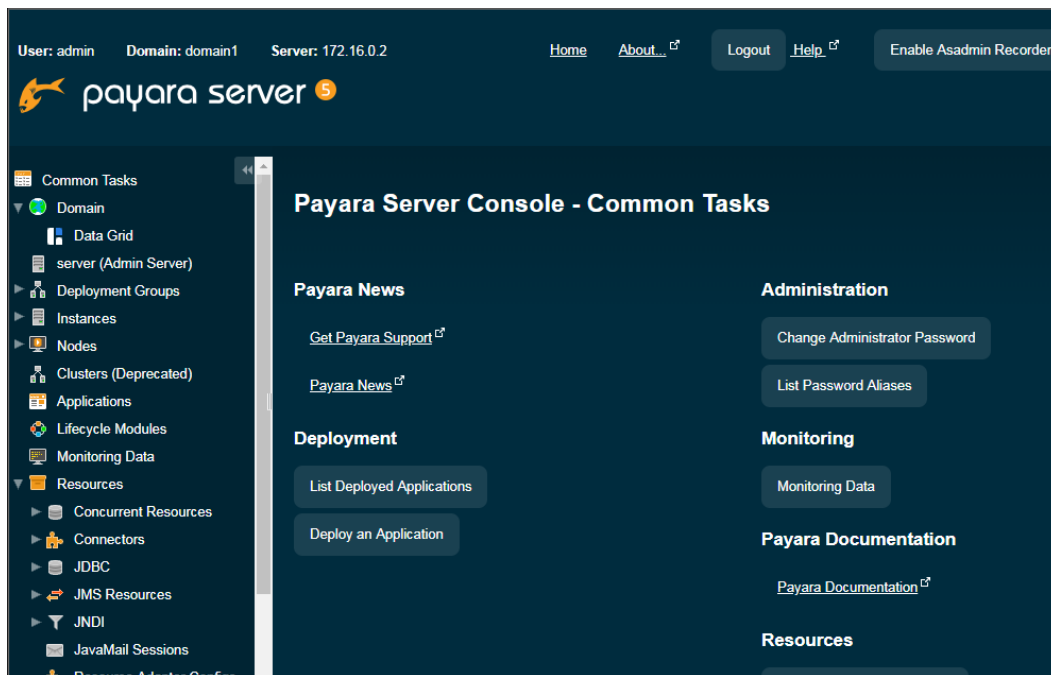
1. DAS ノードとなるサーバを起動します。以下の手順を参考にしてください。

- [Payara の起動と停止](#)

DAS ノードの Payara の管理コンソールにアクセスする

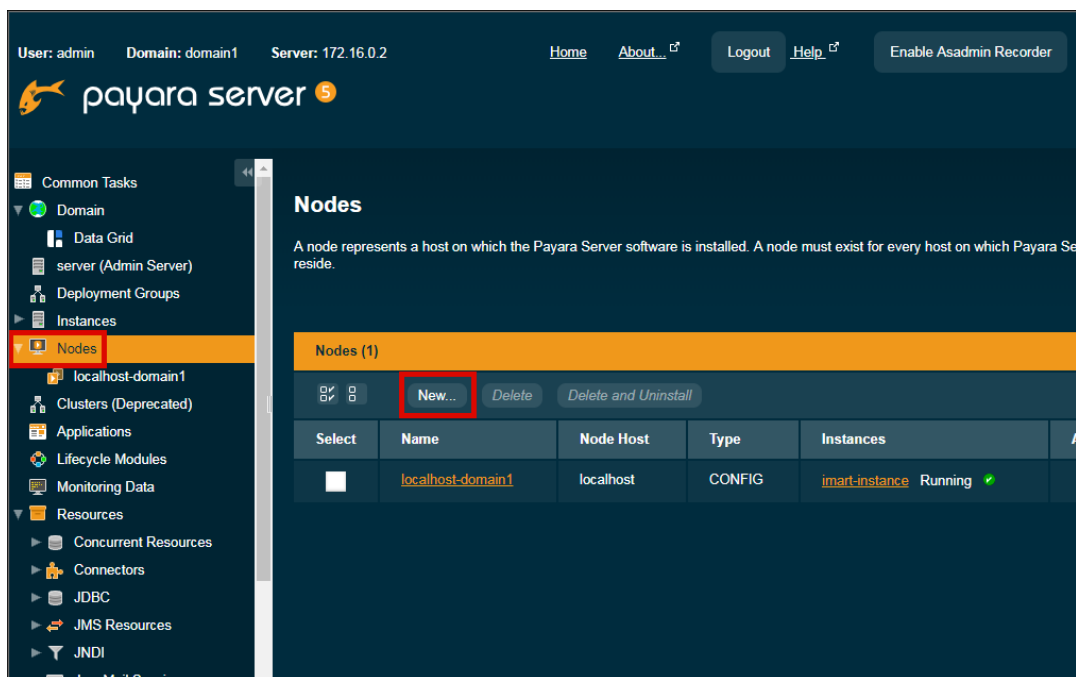
1. DAS ノードとなるサーバの管理コンソールを開きます。以下の手順を参考にしてください。

- [管理コンソールにアクセス](#)



SSH ノードの設定を行う

1. 左ペインのメニューから [Nodes] を選択し、[New] をクリックします。



2. 下記のように入力し、[OK] をクリックします。

Name	ノードの名前を入力します。 例として、ここでは remote-node1 と入力します。
Type	SSH
Node Host	172.16.0.3
Node Directory	未入力で構いません。
Installation Directory	SSH ノードとなるサーバ 上の Payara のインストールパスを入力します。 例として、ここでは /var/payara と入力します。
Install Payara Server	選択しないでください。
Force	選択しないでください。
SSH Port	22

SSH User Name	root
SSH User Authentication	Password
SSH User Password	root ユーザのパスワードを入力します。
Configuration	imart-config を選択してください。 Reference the Selected Configuration を選択してください。



コラム

SSH Port, SSH User Name, SSH User Authentication 等の SSH の設定については、DAS ノードとなるサーバから SSH ノードとなるサーバへのアクセスが可能な SSH の情報を設定してください。公開鍵認証の場合 SSH User Authentication は Password ではなく Key File を選択してください。この場合、Key File には鍵ファイルへのパスを指定してください。



コラム

SSH User Name には、SSH ノード上の Payara ディレクトリ、ストレージディレクトリ等へのアクセス権があるユーザを指定してください。

User: admin Domain: domain1 Server: 172.16.0.2 Home About... Logout Help Enable Asadmin Recorder

payara server 5

Common Tasks

- Domain
- Data Grid
- server (Admin Server)
- Deployment Groups
- Instances
- Nodes**
- Clusters (Deprecated)
- Applications
- Lifecycle Modules
- Monitoring Data
- Resources
- Concurrent Resources
- Connectors
- JDBC
- JMS Resources
- JNDI
- JavaMail Sessions
- Resource Adapter Configs

New Node OK

Create a node on which Payara Server instances can reside.

Name: * remote-node1
Unique name for the node. The name must not contain any reserved words or characters.

Type: SSH
If the type is CONFIG, the node is not enabled for remote communication and the DCOM or SSH information is removed from the page. To create instances on a remote node of type CONFIG, use the create-local-instance subcommand. You cannot use the Administration Console for this.

Node Host: 172.16.0.3
Name of the host that the node represents. If the type is DCOM or SSH, the node host is required.

Node Directory:
Path to the directory that will contain files for instances created on this node. The default is "\${com.sun.aas.productRoot}/glassfish/nodes".

Installation Directory:

User: admin Domain: domain1 Server: 172.16.0.2 Home About... Logout Help Enable Asadmin Recorder

payara server 5

Common Tasks

- Domain
- Data Grid
- server (Admin Server)
- Deployment Groups
- Instances
- Nodes**
- localhost-domain1
- Clusters (Deprecated)
- Applications
- Lifecycle Modules
- Monitoring Data
- Resources
- Concurrent Resources
- Connectors
- JDBC
- JMS Resources
- JNDI
- JavaMail Sessions

Installation Directory: /var/payara
The full path to the parent of the base installation directory of the Payara Server software on the host. For example, /export/payara41.

Install Payara Server: ☒ Enabled
If selected, the Payara Server software on the DAS host is copied to the node host when the node is created.

SSH

Force: ☒ Enabled
Specifies whether the node is created even if validation of the node's parameters fails.

SSH Port: 22
The port to use for SSH connections to this node's host. The default is 22.

SSH User Name: root
The user that is to run the process for connecting to this node's host through SSH. The default user that is running the DAS process.

User: admin Domain: domain1 Server: 172.16.0.2 Home About... Logout Help Enable Asadmin Recorder

SSH

Force: ☒ **Enabled**
Specifies whether the node is created even if validation of the node's parameters fails.

SSH Port:
The port to use for SSH connections to this node's host. The default is 22.

SSH User Name:
The user that is to run the process for connecting to this node's host through SSH. The default user that is running the DAS process.

SSH User Authentication:
Select how the SSH user is authenticated when logging in to this node's host.

SSH User Password:
Type the password that the SSH user will use for logging in to the host that this node represents.

OK

3. SSH ノードを登録できました。

User: admin Domain: domain1 Server: 172.16.0.2 Home About... Logout Help Enable Asadmin Recorder

Nodes

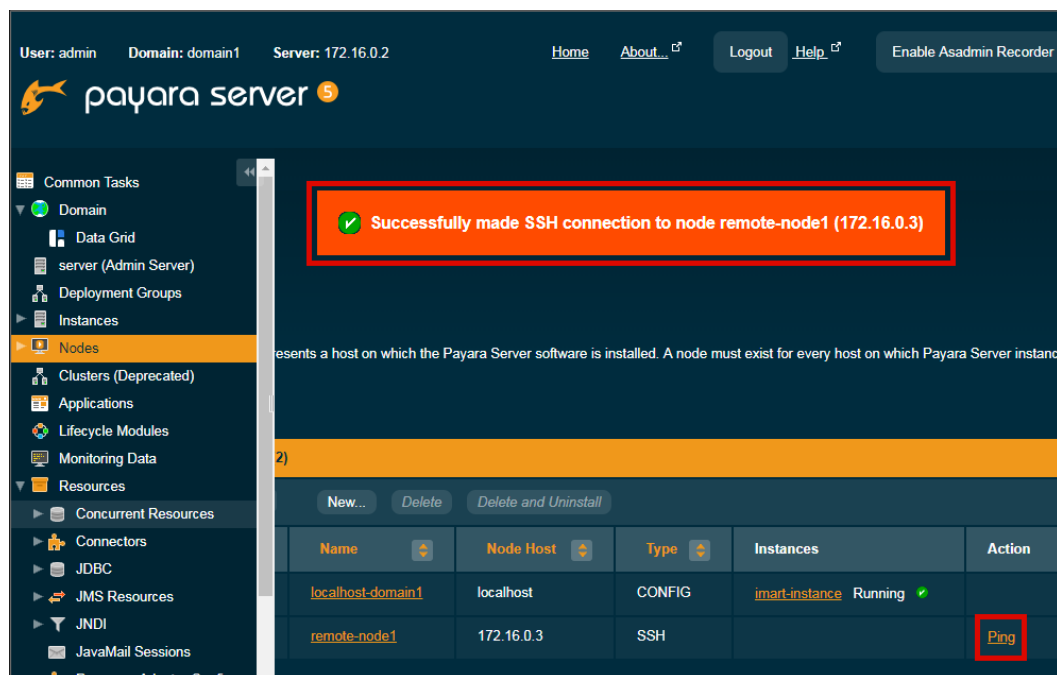
A node represents a host on which the Payara Server software is installed. A node must exist for every host on which Payara Server resides.

Nodes (2)

☒ ☐ New... Delete Delete and Uninstall

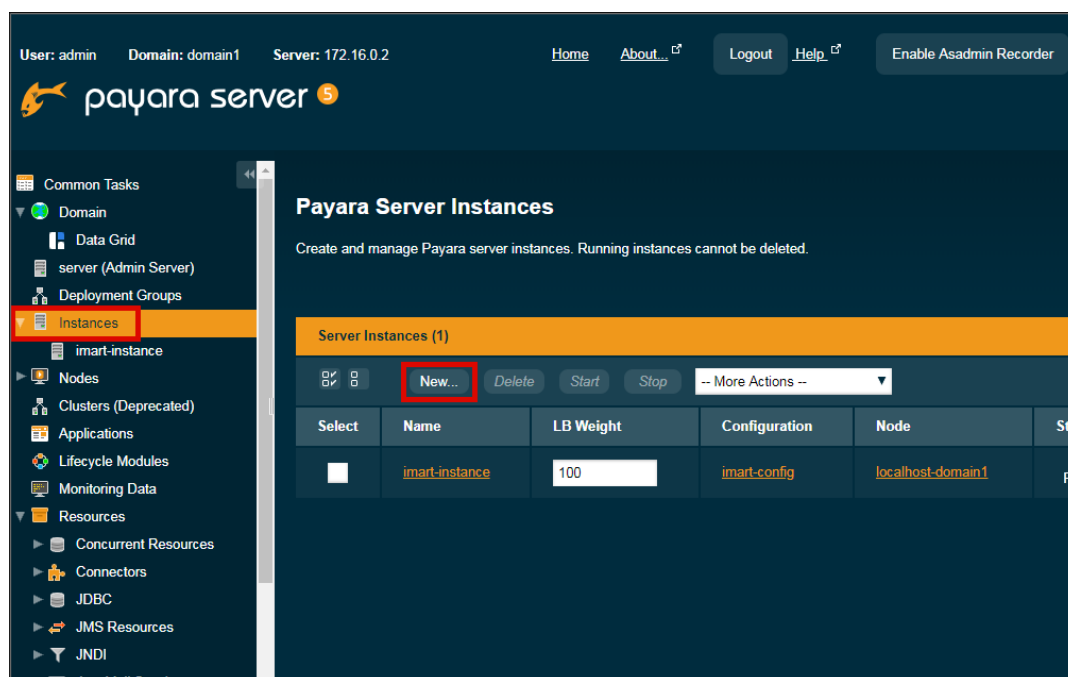
Select	Name	Node Host	Type	Instances
☒	localhost-domain1	localhost	CONFIG	imart-instance Running ✓
☐	remote-node1	172.16.0.3	SSH	

4. [Ping] をクリックすることで、SSH の疎通確認を行えます。



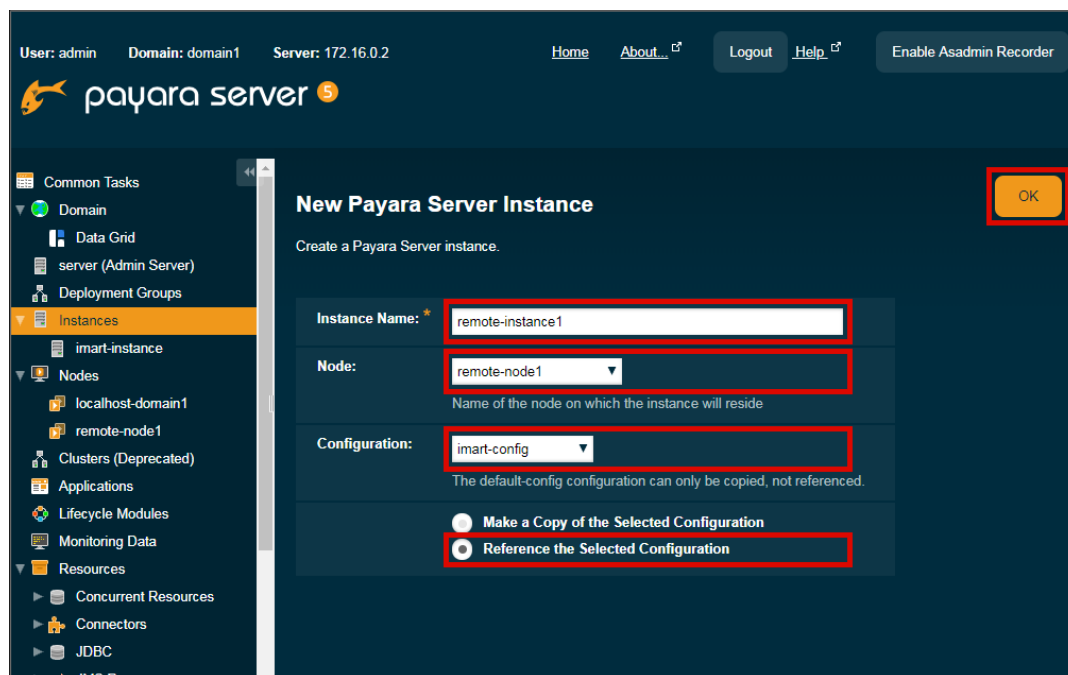
SSH ノードインスタンスを作成する

1. 左ペインのメニューから [Instances] を選択し、[New] をクリックします。

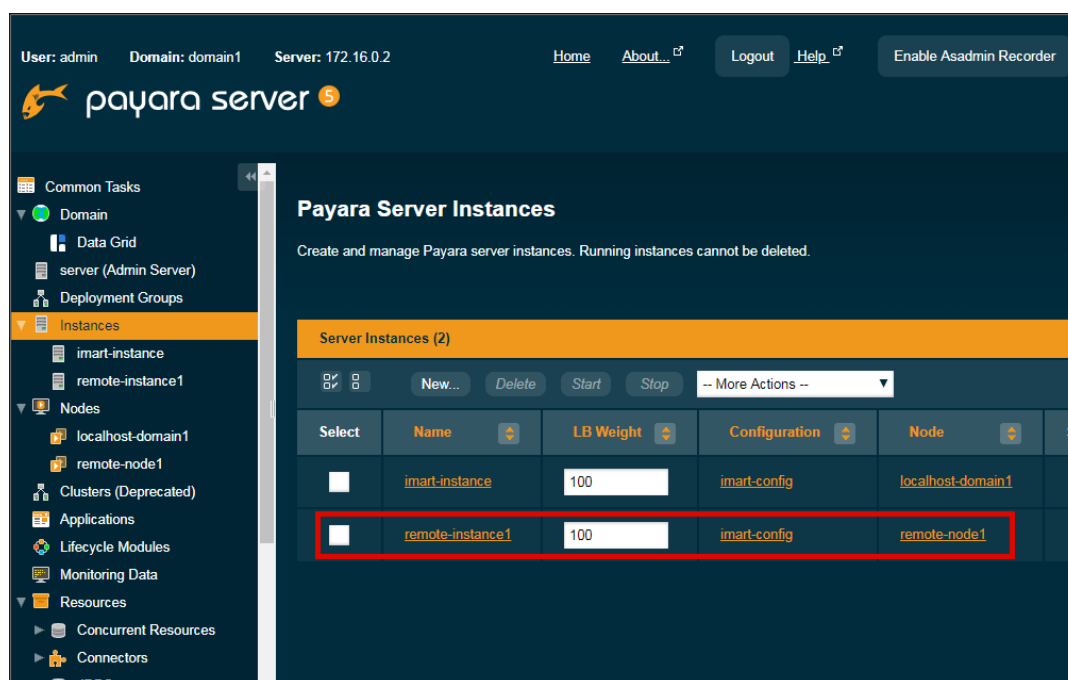


2. 以下のように入力し、[OK] をクリックします。

Instance Name	インスタンスの名前を入力します。 例として、ここでは remote-instance1 と入力します。
Node	remote-node1
Configuration	imart-config を選択してください。 Reference the Selected Configuration を選択してください。

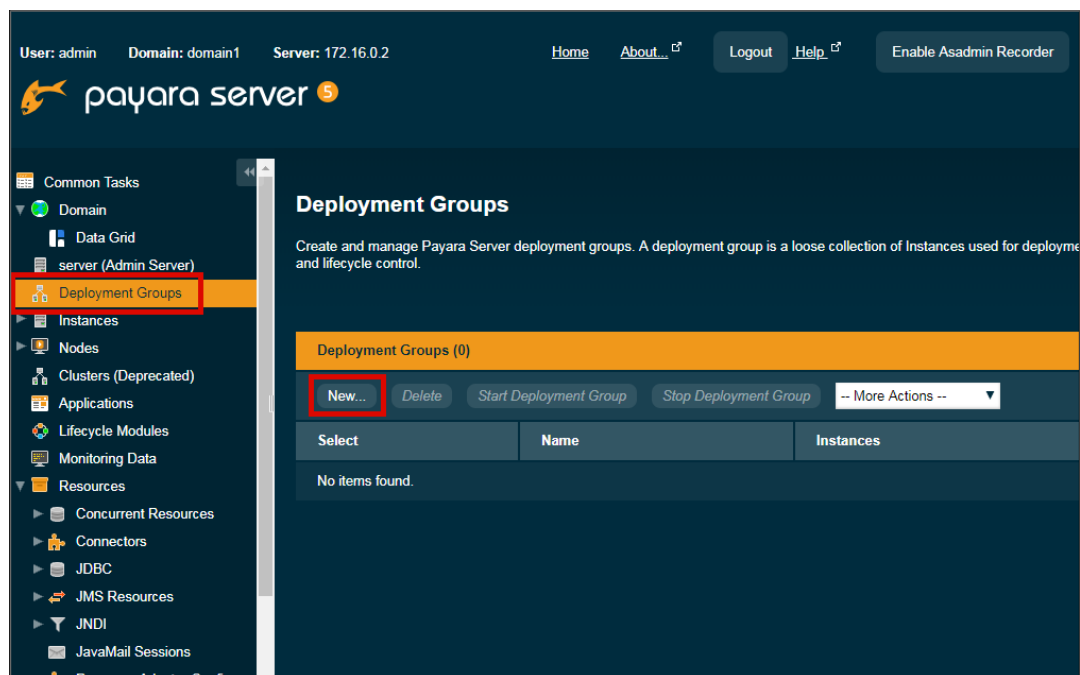


3. インスタンスを登録できました。



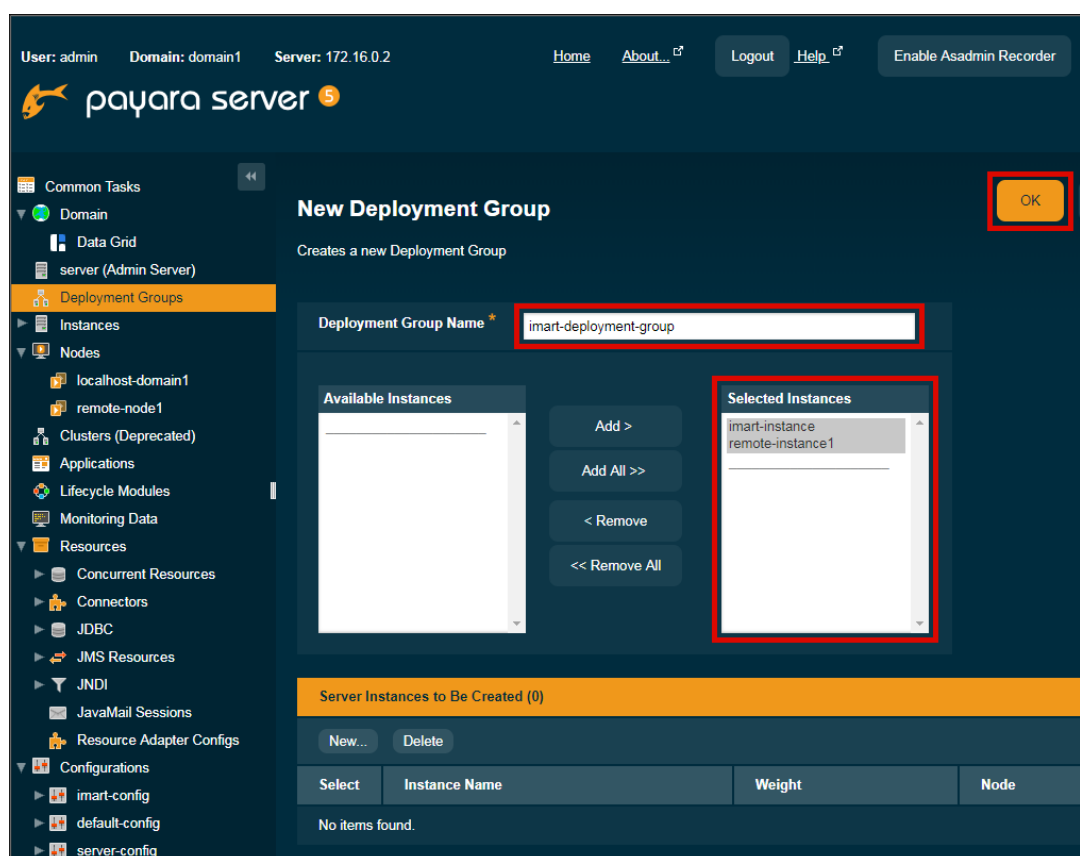
Deployment Group を作成する

1. 左ペインのメニューから [Deployment Groups] を選択し、[New] をクリックします。

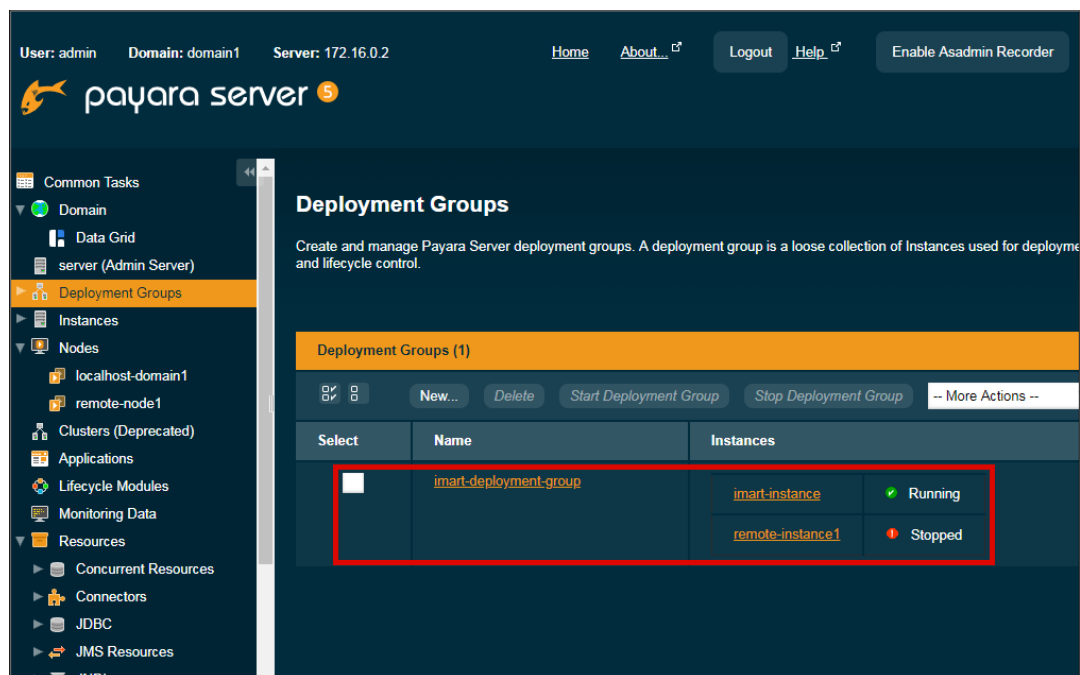


2. 以下のように入力し、[OK] をクリックします。

Deployment Group Name	デプロイメントグループの名前を入力します。 例として、ここでは imart-deployment-group と入力します。
Selected Instances	[imart-instance] と [remote-instance1] を追加します。

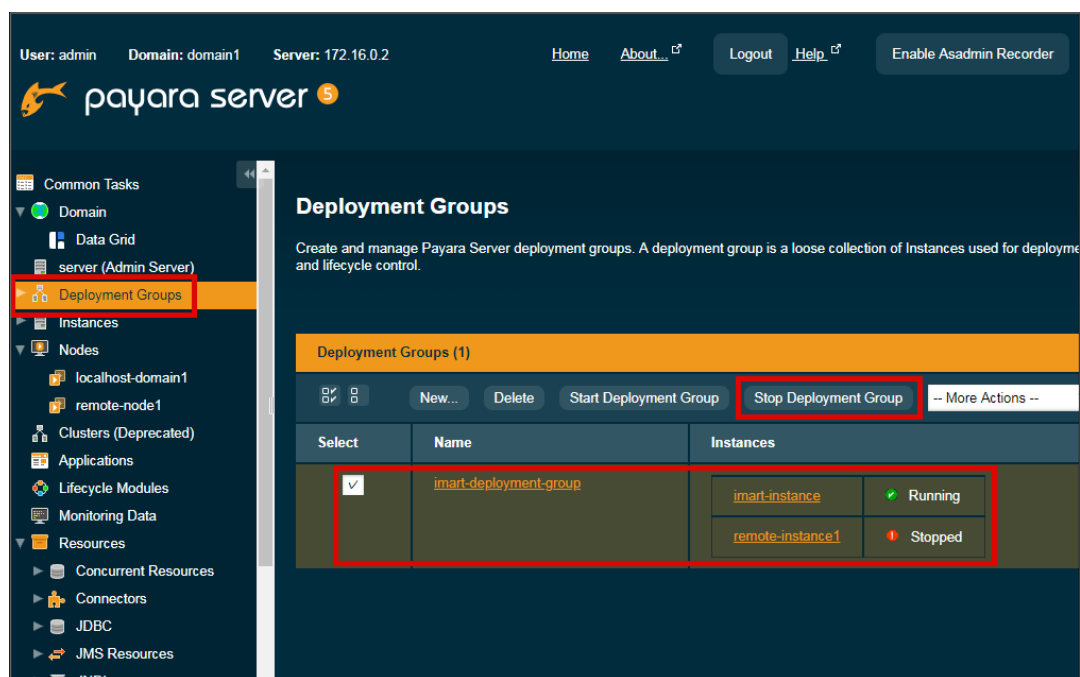


3. デプロイメントグループを登録できました。

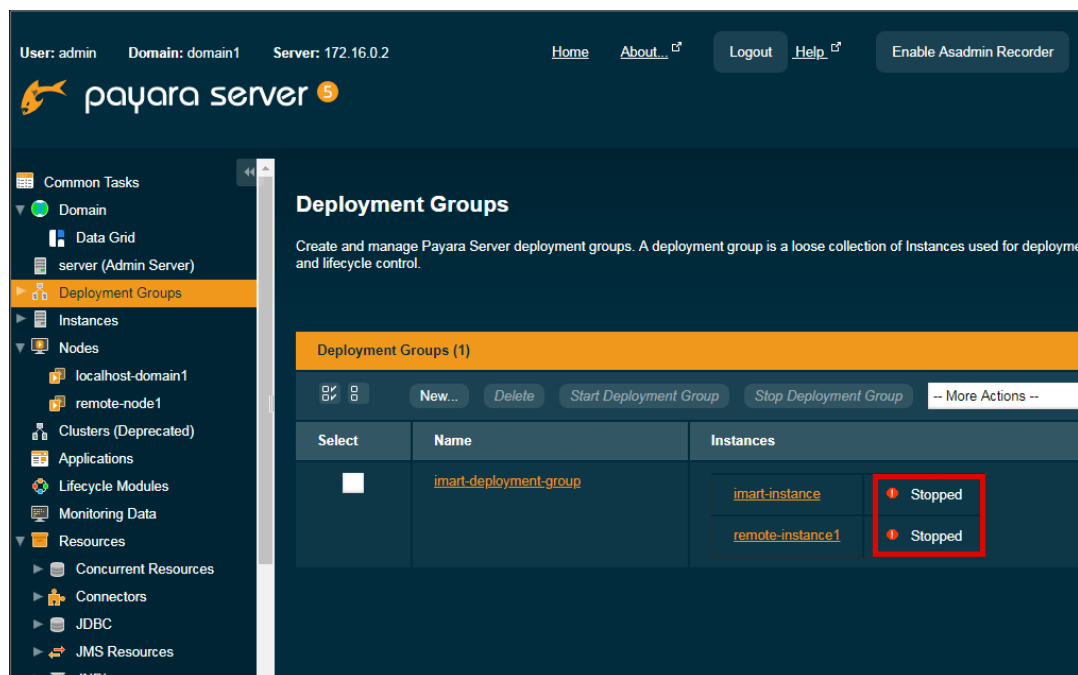


Deployment Group を終了する

1. 左ペインのメニューから [Deployment Groups] を選択し、[imart-deployment-group] を選択し、[Stop Deployment Group] をクリックします。

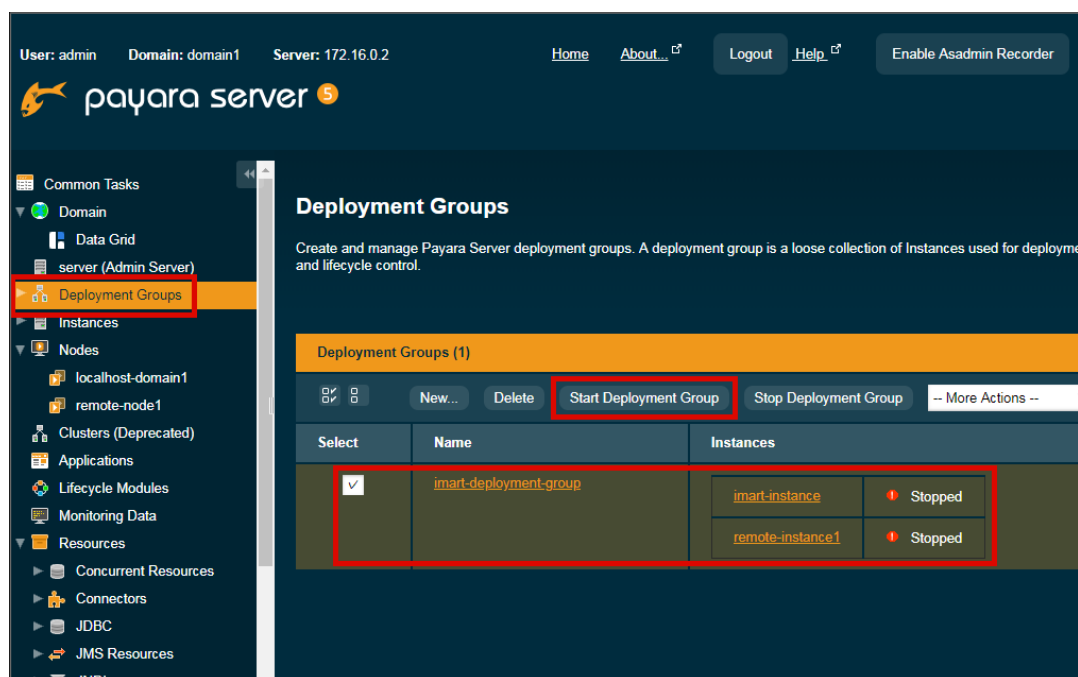


2. デプロイメントグループを終了できました。

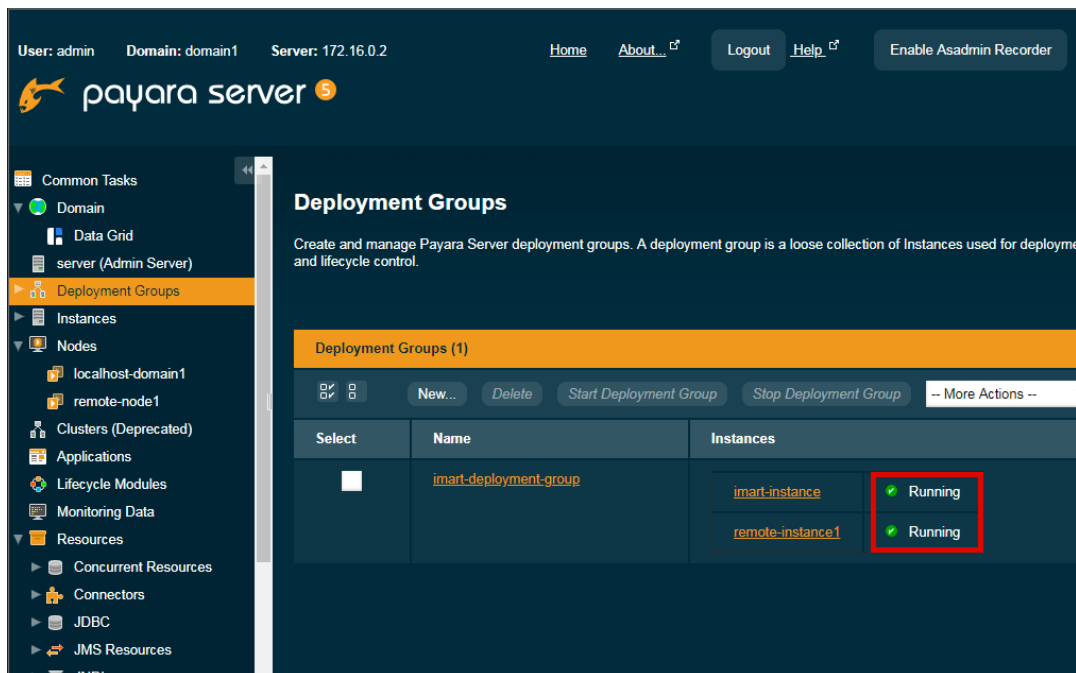


Deployment Group を起動する

1. 左ペインのメニューから [Deployment Groups] を選択し、[imart-deployment-group] を選択し、[Start Deployment Group] をクリックします。

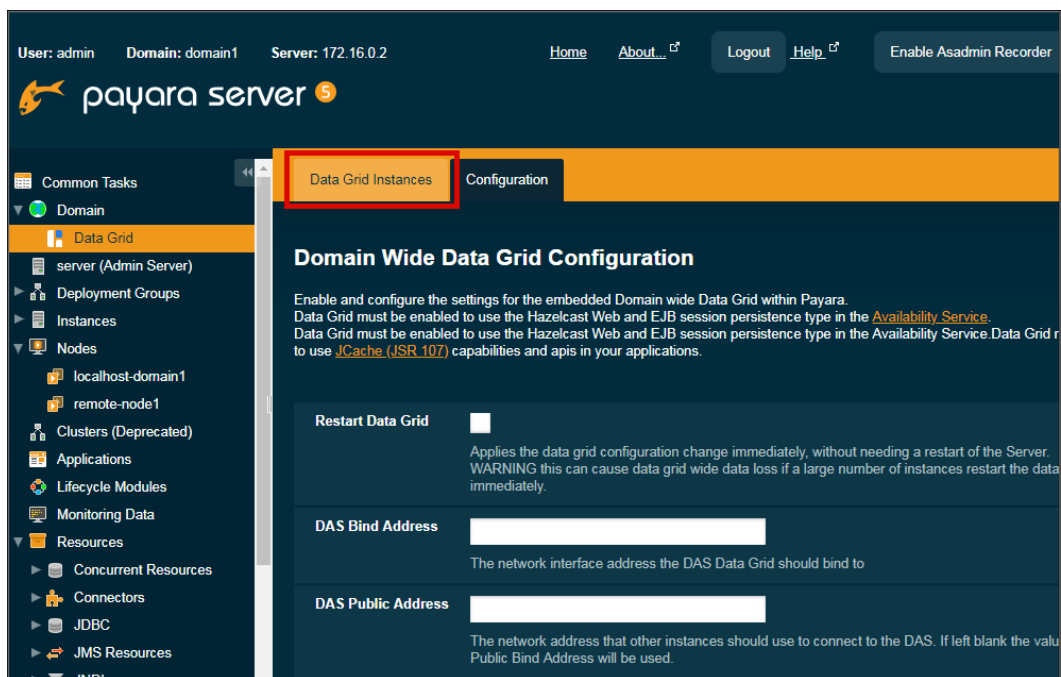


2. デプロイメントグループを開始できました。



Domain Data Grid を確認する

1. 左ペインのメニューから [Domain] - [Data Grid] を選択し、[Data Grid Instances] をクリックします。



2. [imart-instance] と [remote-instance1] がクラスタリングされていることを確認できます。

User: admin Domain: domain1 Server: 172.16.0.2 Home About... Logout Help Enable Asadmin Recorder

payara server 5

Common Tasks Domain Data Grid server (Admin Server) Deployment Groups Instances Nodes localhost-domain1 remote-node1 Clusters (Deprecated) Applications Lifecycle Modules Monitoring Data Resources Concurrent Resources Connectors JDBC JMS Resources

Data Grid Instances Configuration

Data Grid Instances

A list of the Data Grid Instances visible to this domain

All Data Grid Instances (3)

Name	Type	Group	Host	HTTP Ports	HTTPS Ports	Admin Port
remote-instance1	INSTANCE	imart-config	/172.16.0.3	28081	28182	24849
imart-instance	INSTANCE	imart-config	/172.16.0.2	28080	28181	24848
server	DAS	server-config	/172.16.0.2	8080	8181	4848

User: admin Domain: domain1 Server: 172.16.0.2 Home About... Logout Help Enable Asadmin Recorder

payara server 5

Common Tasks Domain Data Grid server (Admin Server) Deployment Groups Instances Nodes localhost-domain1 remote-node1 Clusters (Deprecated) Applications Lifecycle Modules Monitoring Data Resources Concurrent Resources Connectors JDBC JMS Resources

Payara Server Instances (3)

Instance Name	Group	Host Name	HTTP Ports	HTTPS Ports	Admin Port
remote-instance1	imart-config	/172.16.0.3	28081	28182	24849
imart-instance	imart-config	/172.16.0.2	28080	28181	24848
server	server-config	/172.16.0.2	8080	8181	4848

Payara Micro Instances (0)

Command: Send Asadmin Command

Select	Instance Name	Group	Host Name	HTTP Ports	HTTPS Ports	Admin Port	Hazelcast Port	Lite Me
No Micro instances in the cluster								

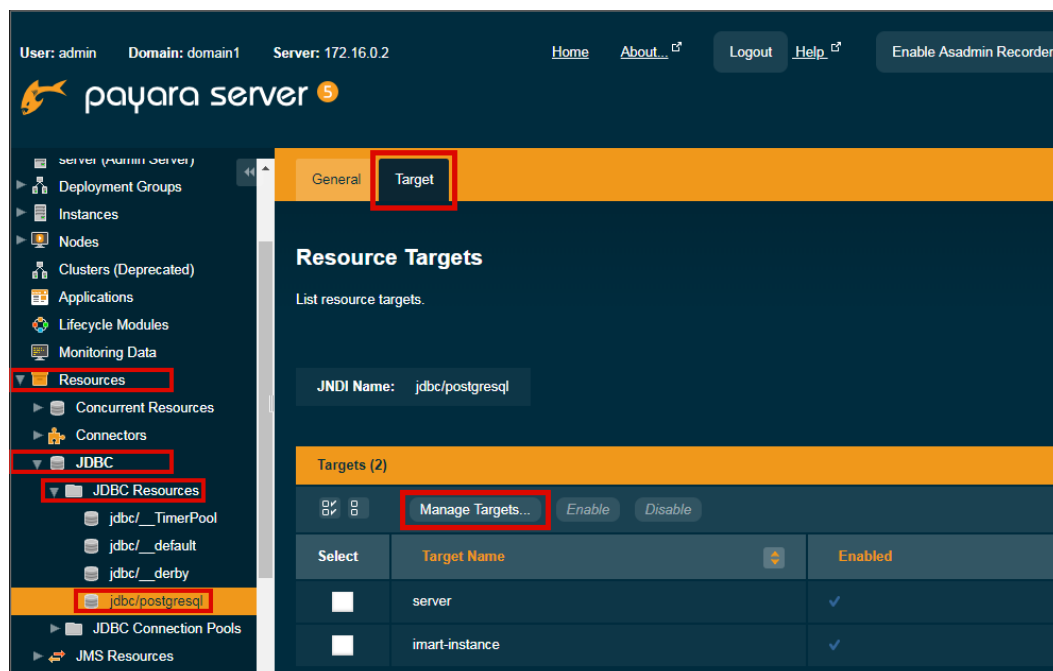


コラム

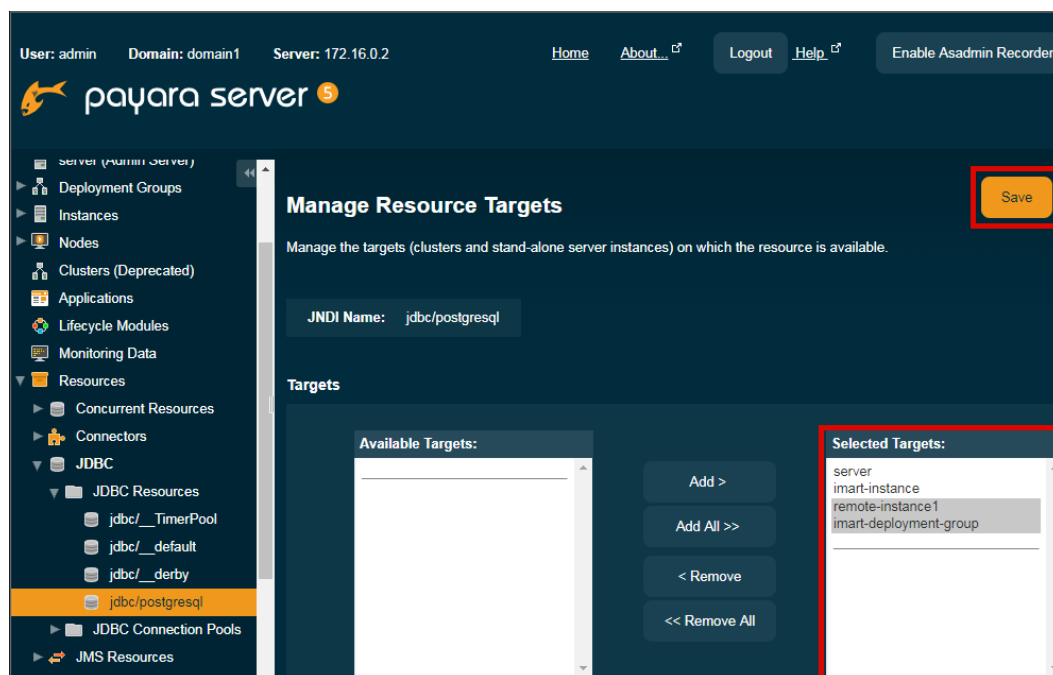
Payara 5 の Domain Data Grid 機能として、DAS ノード (Instance Name=server) もクラスタリングに参加します。

デプロイメントグループを JDBC リソースの対象に設定する

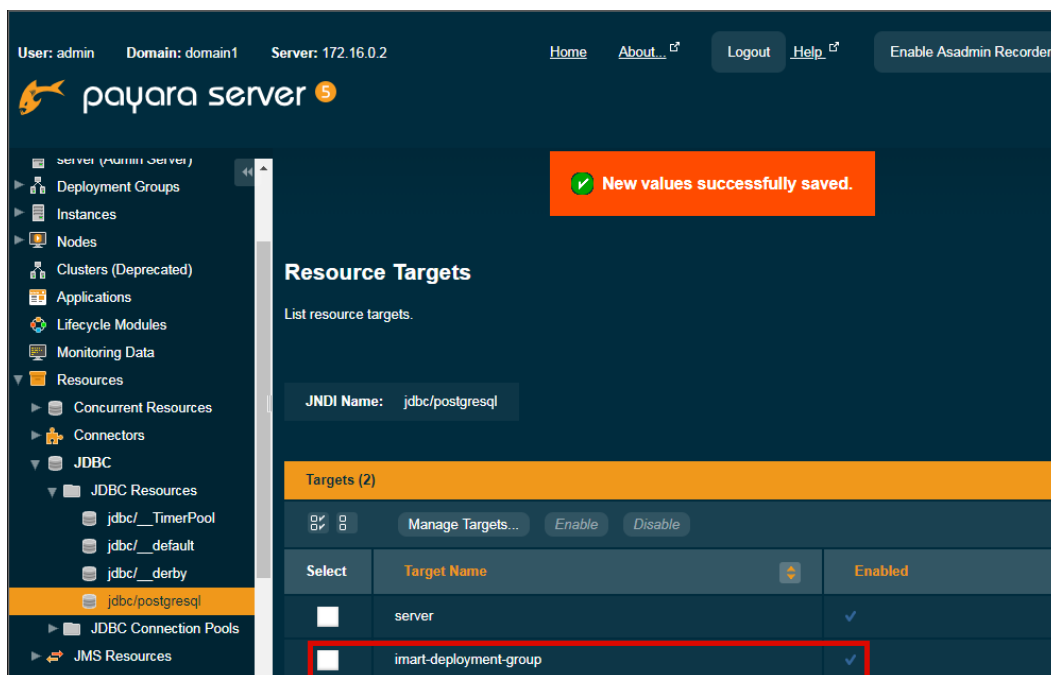
1. 左ペインのメニューから [Resources] - [JDBC] - [JDBC Resources], 対象となる JDBC リソースをクリックします。例として、ここでは jdbc/postgresql を選択します。選択後、[Target] - [Manage Targets] をクリックします。



2. [imart-deployment-group] を [Selected Targets] に追加します。その他は必要に応じて追加します。追加後、[Save] をクリックします。

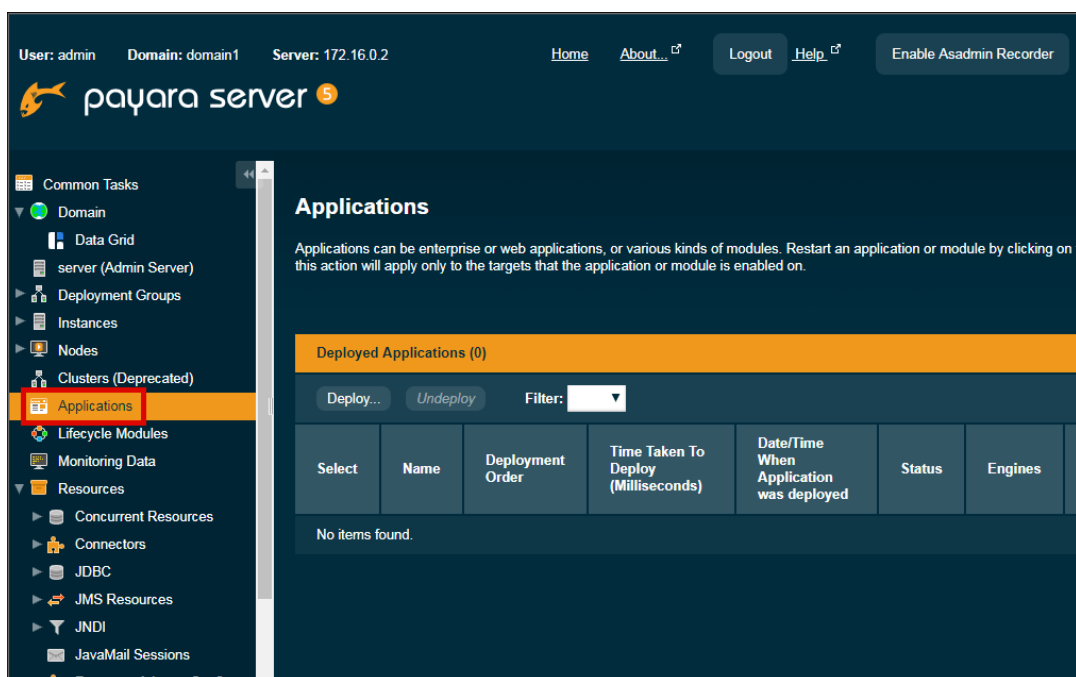


3. デプロイメントグループを JDBC リソースの対象に追加できました。

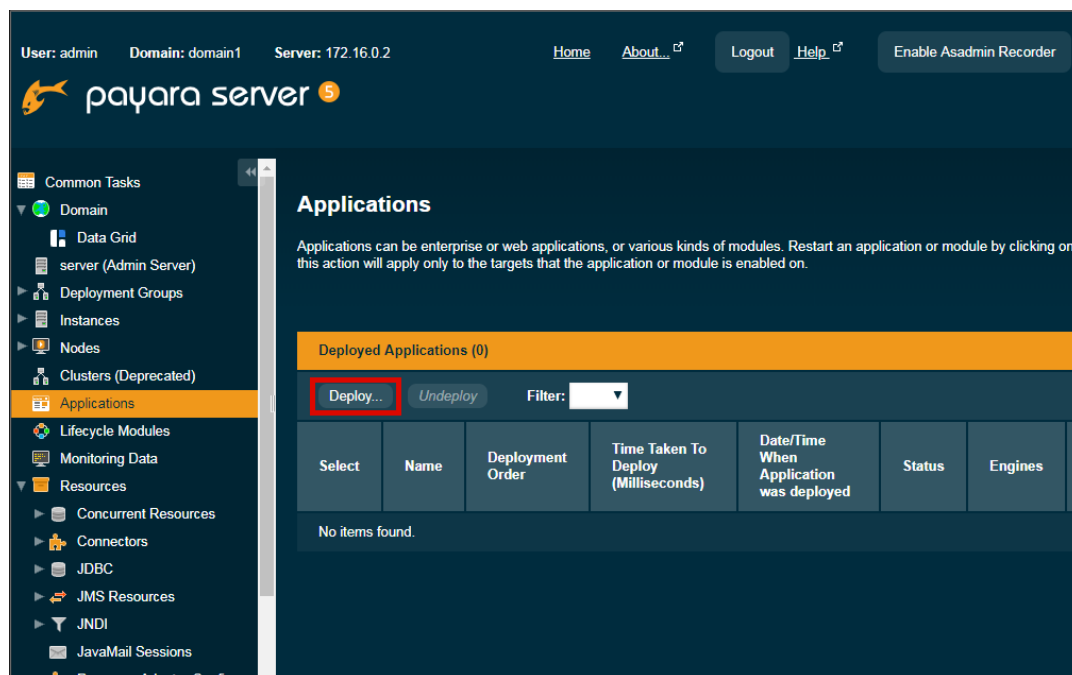


アプリケーションをデプロイする

1. 左ペインのメニューから [Applications] をクリックします。

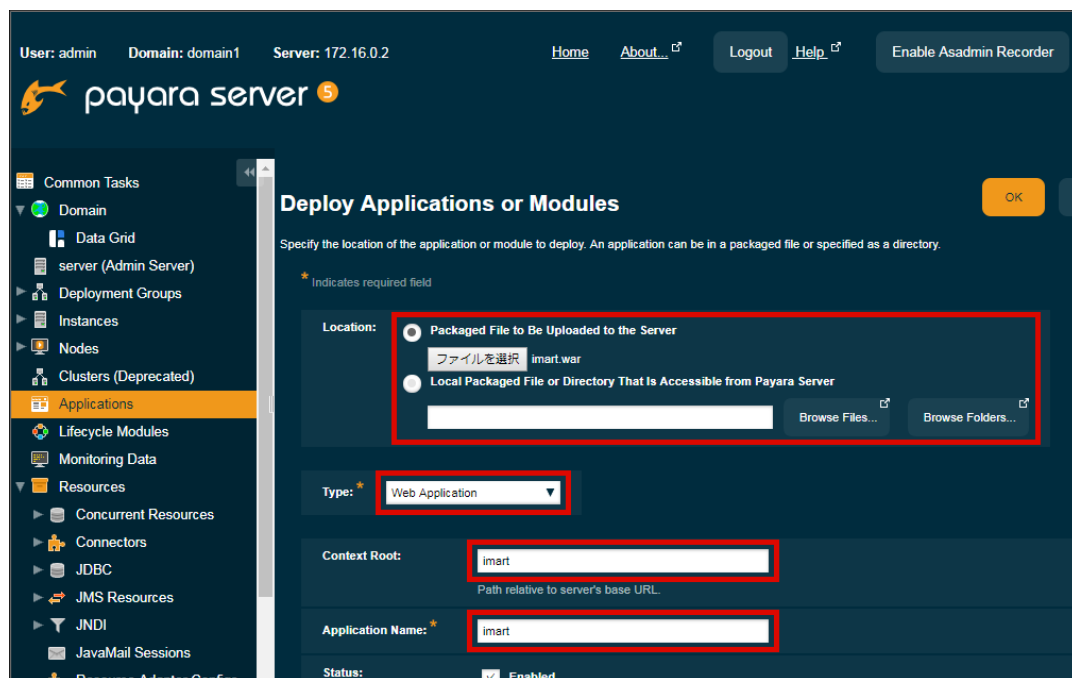


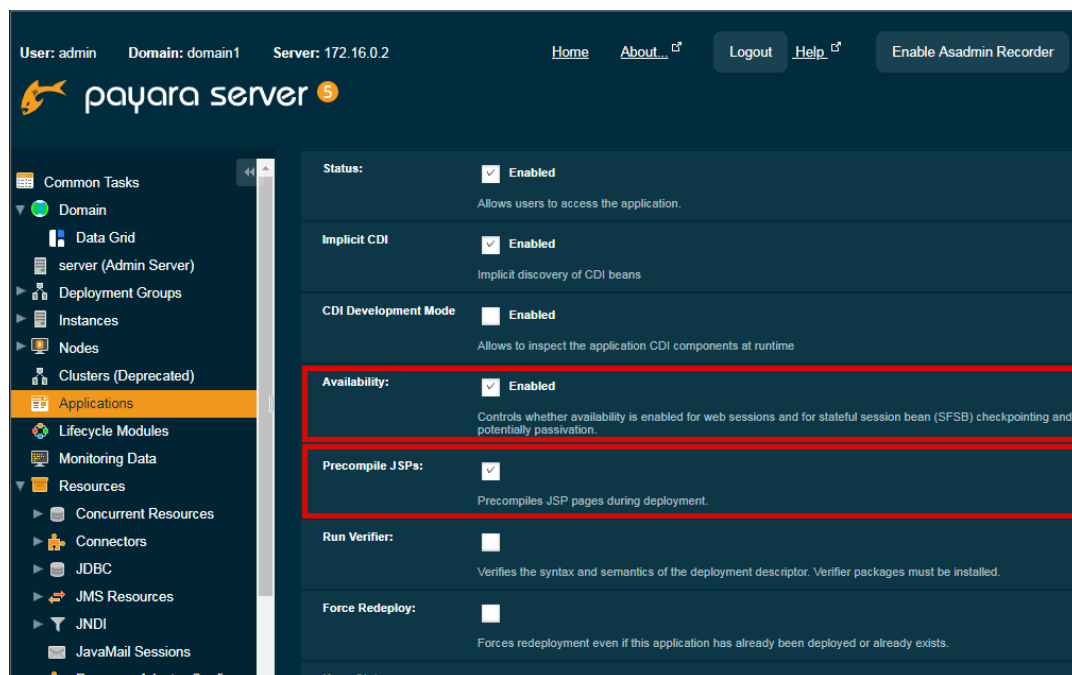
2. [Deploy] をクリックします。



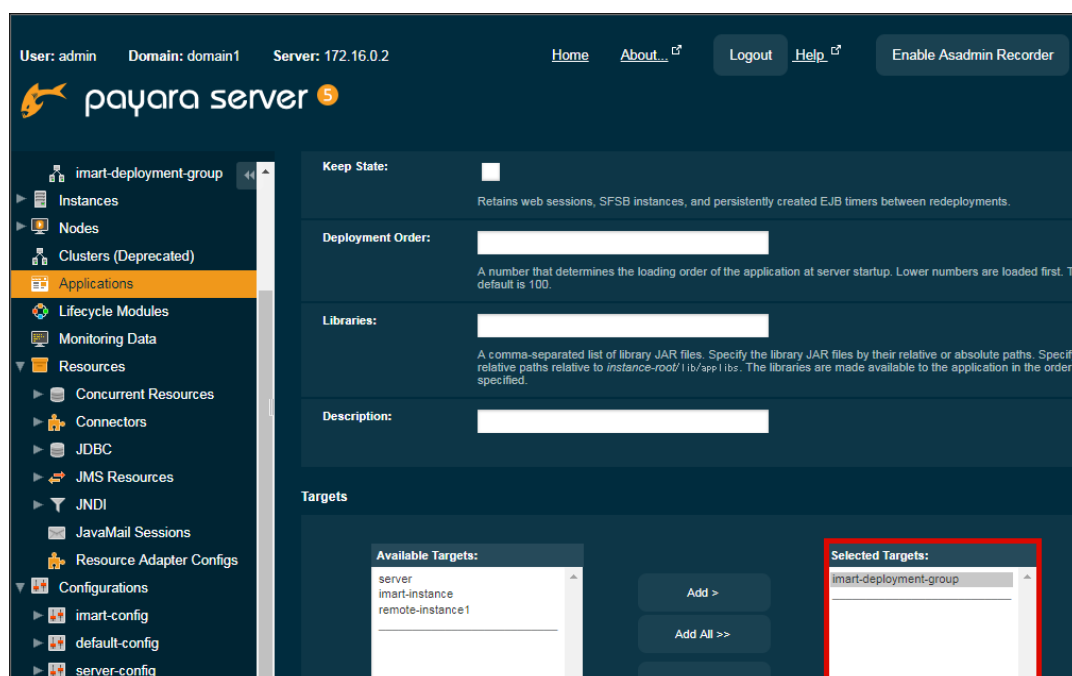
3. 以下のような入力します。

Location	[ファイルを選択] より、war ファイルを選択してアップロードします。 または、war ファイルを DAS ノードとなるサーバに別途アップロードし、Browse Files よりアップロードした war ファイルを選択します。
Type	Web Application を選択します。
Context Root	コンテキストルート名を入力します。例として、ここでは imart を設定します。
Application Name	アプリケーション名を入力します。例として、ここでは imart を設定します。
Availability	選択します。Availability を有効化しない場合、セッションレプリケーションを行えません。
Precompile JSPs	選択します。

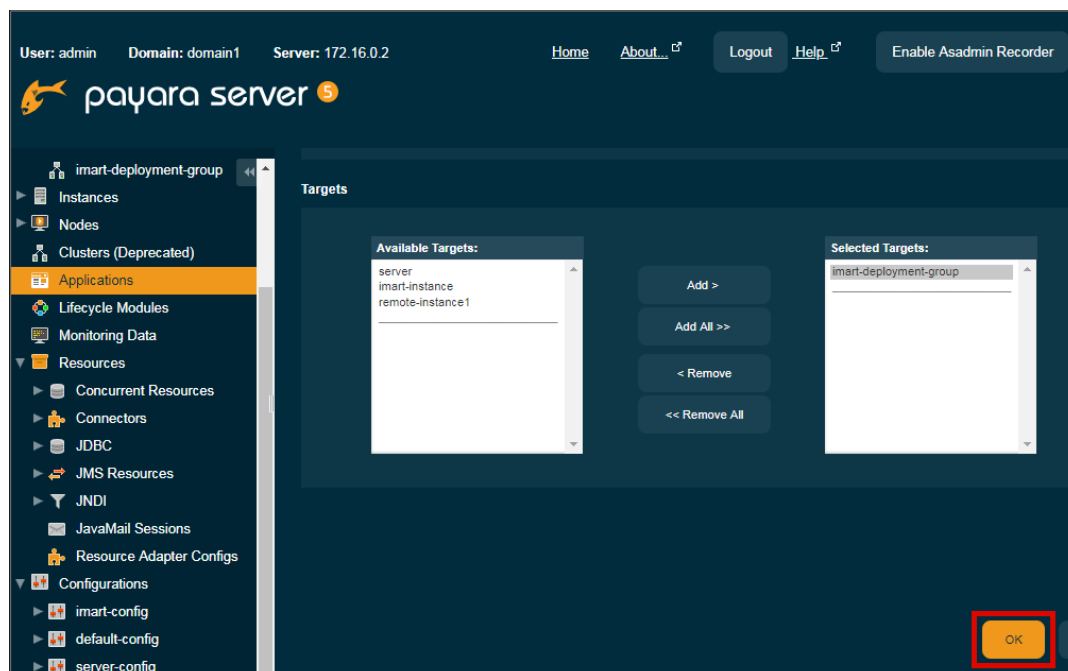




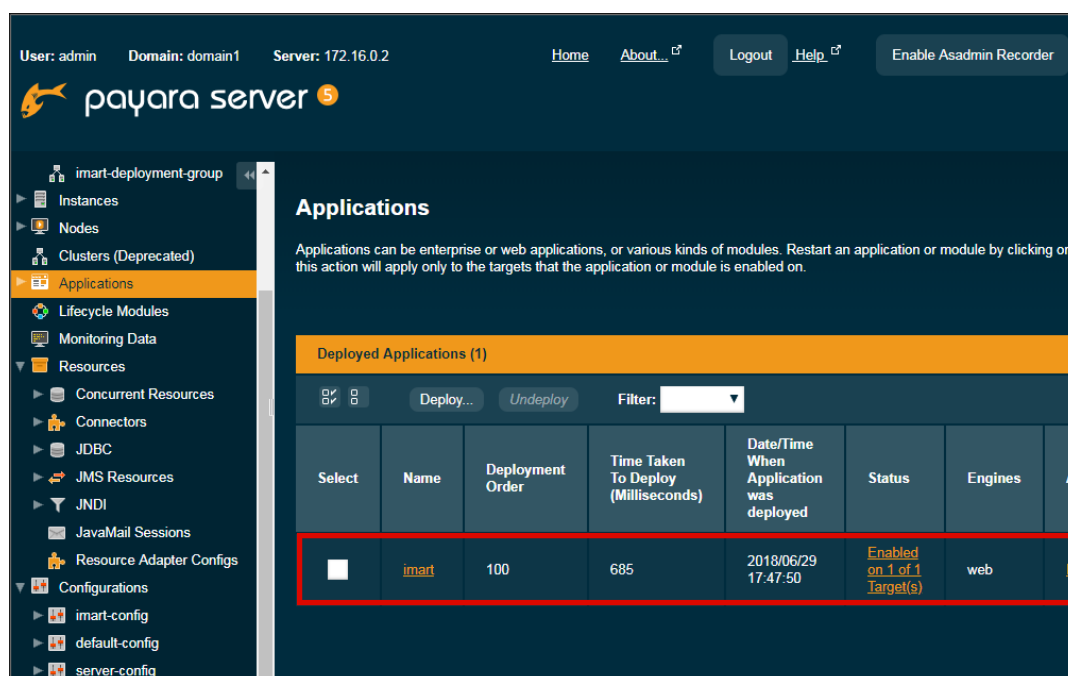
4. [Selected Targets] に [imart-deployment-group] を追加します。



5. [OK] をクリックし、デプロイします。



6. デプロイできました。



インスタンスの起動確認

1. 「インスタンスの起動確認」と同様の手順でインスタンスが起動しているポートを確認します。

The screenshot shows the Payara Server Administration Console. The left sidebar has a tree view with 'imart-instance' selected under 'Instances'. The main panel shows the 'System Properties' tab for 'imart-instance'. The 'Instance Name' is 'imart-instance'. Under 'Additional Properties (11)', the 'HTTP_LISTENER_PORT' is set to '28080' in the 'Current Value' column. The 'ASADMIN_LISTENER_PORT' is set to '24848'. A red box highlights the 'HTTP_LISTENER_PORT' row.



コラム

DAS ノードとなるサーバ上で 28080 ポートで起動していることが確認できます。

The screenshot shows the Payara Server Administration Console. The left sidebar has a tree view with 'remote-instance1' selected under 'Instances'. The main panel shows the 'System Properties' tab for 'remote-instance1'. The 'Instance Name' is 'remote-instance1'. Under 'Additional Properties (11)', the 'HTTP_LISTENER_PORT' is set to '28081' in the 'Current Value' column. The 'ASADMIN_LISTENER_PORT' is set to '24849'. A red box highlights the 'HTTP_LISTENER_PORT' row.



コラム

SSH ノードとなるサーバ上で 28081 ポートで起動していることが確認できます。

2. 上記の手順で確認できた以下の二つの URL に対してロードバランスを行ってください。

- <http://172.16.0.2:28080/imart>
- <http://172.16.0.3:28081/imart>



注意

Payara のクラスタリング環境を運用する場合は、必ず Web Server やロードバランサをスティッキーセッションとし、同一セッションのリクエストが同じ Payara に送信されるように設定してください。

Apache Solr

Apache Solr は、「IM-ContentsSearch」を利用するために必要です。



コラム

プロジェクトの作成とモジュールの選択で IM-ContentsSearch アプリケーションを選択しない場合、Apache Solr のセットアップは不要です。

- [セットアップで困ったら・・・](#)
- [アップデート パッチの適用](#)