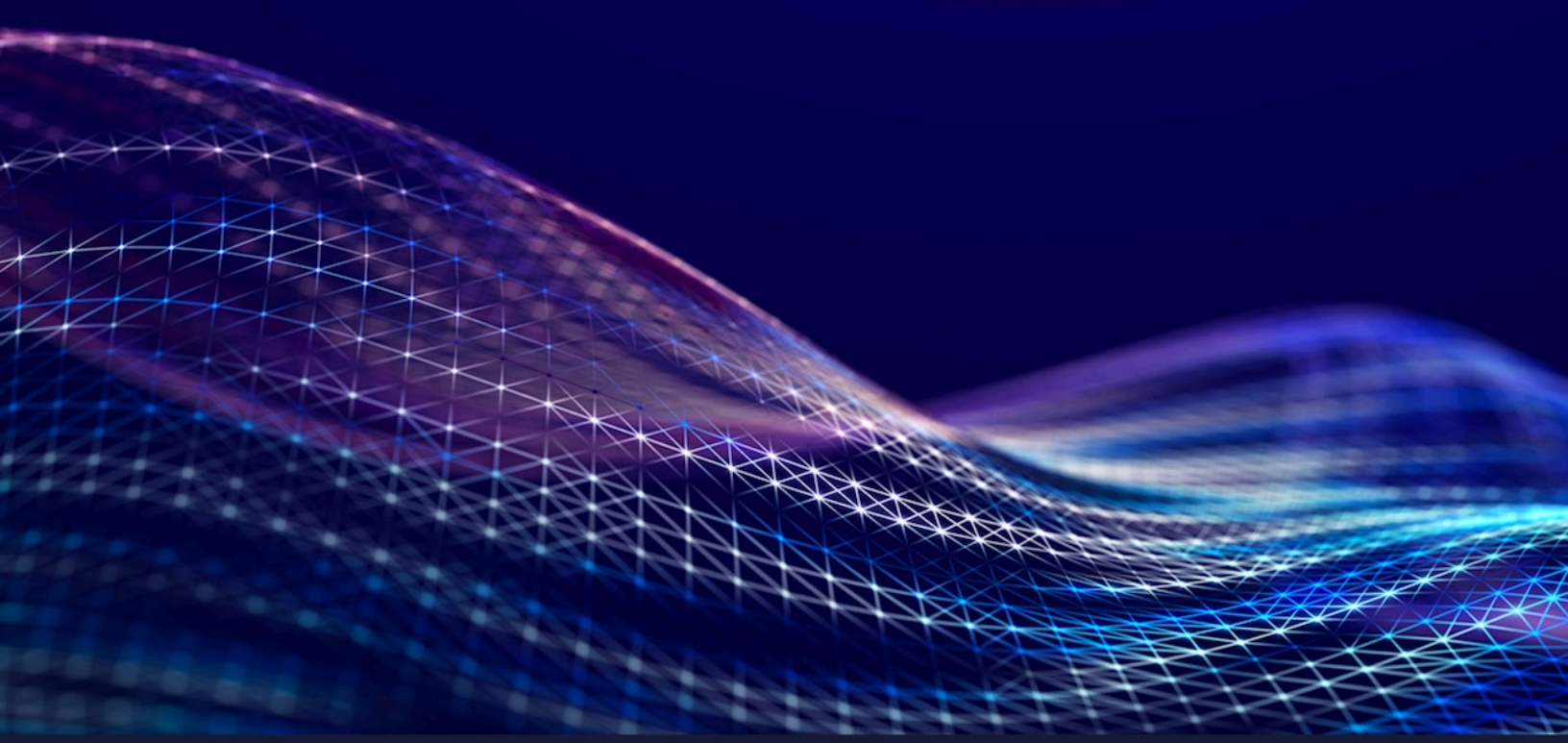


Tungsten RPA
管理者ガイド
2026.1

TUNGSTEN AUTOMATION

目次

はじめに

本ガイドは、エンタープライズ環境で Tungsten RPA を展開するシステム管理者を対象としています。

関連ドキュメント

Tungsten RPA のドキュメント セットは、次の場所から入手できます。

https://docshield.tungstenautomation.com/Portal/Products/RPA/2026.1-cqvh2o7vk9/RPA.htm

完全なドキュメント セットにオンラインでアクセスするには、インターネットに接続する必要があります。

インターネットに接続せずにアクセスする場合は、『Tungsten RPA インストール ガイド』の「オフライン ドキュメント」を参照してください。

ドキュメント セットには、次のようなリソースがアルファベット順で含まれています。

***Tungsten RPA* 管理者ガイド**
Tungsten RPA での管理タスクについて説明します。

***Tungsten RPA* のベストプラクティス ガイド**
Tungsten RPA 環境でロボット ライフサイクル マネジメントを使用しながらパフォーマンスを最適化し、成功を確実にするために推奨される方法とテクニックを提供します。

***Tungsten RPA Desktop Automation* サービス ガイド**
リモート コンピューターで Desktop Automation を使用するために必要な Desktop Automation サービスを設定および管理する方法について説明します。

***Tungsten RPA* 開発者ガイド**
Java、.NET、および REST API を使用して Management Console 上でロボットを実行するための情報と手順が記載されています。

***Tungsten RPA* ロボット構築スタート ガイド**
Tungsten RPA を使用したロボット構築処理を示すチュートリアルが記載されています。

***Tungsten RPA* のヘルプ**
Tungsten RPA の使用方法について説明しています。ヘルプは Tungsten RPA 製品内から利用できます。

***Tungsten RPA* インストール ガイド**
Tungsten RPA およびそのコンポーネントを開発環境にインストールする方法について説明します。

***Tungsten RPA Java API documentation (Tungsten RPA Java API* ドキュメント)**
開発者が Tungsten RPA で使用できる Tungsten RPA Java API パッケージおよびクラスへのアクセスを提供します。

Tungsten RPA API は、元の製品名である「RoboSuite」に対する詳細な参照を含んでいます。RoboSuite という名前は下位互換性を確保するためにそのまま使用されていいます。API ドキュメントの中では、RoboSuite という用語は Tungsten RPA と同じ意味で使われています。

***Tungsten RPA* リリース ノート**
他の Tungsten RPA ドキュメントからは入手できない最新の詳細やその他の情報が含まれています。

***Tungsten RPA* 技術仕様**
サポートされるオペレーティング システムおよびその他のシステム要件に関する情報が含まれています。

***Tungsten RPA* アップグレード ガイド**
Tungsten RPA やそのコンポーネントを新しいバージョンにアップグレードする手順が含まれています。

***Tungsten RPA* ユーザー ガイド**
Tungsten RPA とそのコンポーネントの使用手順が記載されています。Tungsten RPA のヘルプ のトピックとともに、ヘルプには含まれていない詳細な内容が記載されています。

システム要件

サポートされるオペレーティング システムおよびその他のシステム要件については、以下の Tungsten RPA 製品ドキュメント サイトの『RPA 技術仕様』ドキュメントを参照してください。

トレーニング

Tungsten Automation は、製品を最大限に活用できるように、オンデマンド トレーニングおよびインストラクター主導のトレーニングを提供しています。トレーニング コースとスケジュールの詳細については、Tungsten Automation Learning Cloud を参照してください。

Tungsten RPA ドキュメント サイトには、コンポーネントの理解および RPA でのロボット作成に関する基礎の確認のためのチュートリアルが用意されています。https://docshield.tungstenautomation.com/Portal/Products/RPA/2026.1-cqvh2o7vk9/RPA.htm を参照してください。

Tungsten Automation 製品のヘルプの入手

[Tungsten Automation Knowledge Portal (Tungsten Automation ナレッジ ポータル)] リポジトリにある記事の内容は定期的に更新され、Tungsten Automation 製品の最新情報について参照することができます。製品に関してご不明の点がある場合は、Knowledge Portal (ナレッジ ポータル) で情報を検索することをお勧めします。

[Tungsten Automation Knowledge Portal] にアクセスするには次のリンクを使用してください。

https://knowledge.tungstenautomation.com/

Knowledge Portal は Google Chrome、Mozilla Firefox、または Microsoft Edge 向けに最適化されています。

Knowledge Portal では次のような機能を利用できます。

- 強力な検索機能で必要な情報をすぐに見つけることができます。

  [Search (検索)] ボックスに目的の語句を入力し、検索アイコンを選択してください。
- 製品情報、設定の詳細、リリース情報などのドキュメント。

  記事を見つけるには、Knowledge Portal のホームページにアクセスし、製品に該当するソリューション ファミリーを選択するか、[View All Products (すべての製品を表示)] ボタンを選択します。

Knowledge Portal のホームページからは、次の操作を実行できます。

- Tungsten Automation Community (Tungsten Automation コミュニティ) へのアクセス (全カスタマー)。

  [Tungsten Automation Resources (Tungsten Automation リソース)] メニューで、**[Community (コミュニティ)]** リンクを選択します。
- Tungsten Automation Customer Portal (Tungsten Automation カスタマー ポータル) へのアクセス (一部のカスタマーのみ)。

  [Support Portal Information (サポート ポータルの情報)] ページに移動し、**[Log in to the Tungsten Automation Customer Portal (Tungsten Automation カスタマー ポータルにログイン)]** を選択します。
- Tungsten Automation Partner Portal (Tungsten Automation パートナー ポータル) へのアクセス (一部のパートナーのみ)。

  [Support Portal Information] ページに移動し、**[Log in to the Tungsten Automation Partner Portal (Tungsten Automation パートナー ポータルにログイン)]** を選択します。
- サポート コミットメント、ライフサイクル ポリシー、電子フルフィルメントの詳細、およびセルフサービス ツールへのアクセス。

  [Support Details (サポートの詳細)] ページに移動し、適切な記事を選択します。

第 1 章

概要

Tungsten RPA は、アプリケーション、システム、環境全体で反復的なルールベースのビジネス タスクを自動化するように設計された、エンタープライズ グレードのロボット プロセス自動化プラットフォームです。組織に対して、ソフトウェア ロボットの構築、展開、運用、および監視のためのスケーラブルなフレームワークを提供します。管理者は運用環境を管理し、安全な設定を確保して、自動化されたプロセスの信頼性の高い実行を維持します。

Tungsten RPA は、エンドツーエンドの自動化を実現するために連携して動作するいくつかの主要コンポーネントで構成されています。

- Management Console
- RoboServer
- Kapplets
- ロボット ファイル システム:
- オプションのサービス (Desktop Automation サービス、Synchronizer サービスなど)

Tungsten RPA は柔軟性が高く、従来のオンプレミス インストールから完全にコンテナ化されたクラウド展開までの、さまざまなインフラストラクチャ戦略に適合する複数の展開モデルをサポートします。

- コンテナベースの展開
- Windows サービスとしての展開
- Windows でのサイレント インストールとアンインストール

以下の章では、それぞれのコンポーネントを完全かつ機能的に展開するためのセットアップ方法と管理方法について説明します。

第 2 章

Management Console

Management Console は、Tungsten RPA を管理および操作するために使用する一元的な Web アプリケーションです。Management Console には、システムの動作の設定、ロボットとその実行の管理、自動化のスケジューリング、ログとパフォーマンス データの確認、ユーザーと権限の管理を行うためのツールが用意されています。すべての主要な運用機能は、RPA の展開全体の制御層として機能する Management Console を通過します。

Tungsten RPA には、Management Console の実行に必要な、本番環境対応の Java ランタイム環境と組み込みの Apache Tomcat ライブラリが含まれています。

デフォルトでは、Management Console は非本番環境向けの H2 データベースを使用します。本番環境での展開では、Management Console を本番レベルのデータベースに接続するように設定し、必要な接続とセキュリティの設定を完了します。

Management Console をセットアップおよび管理するには、次のトピックを参照してください。

- Management Console の起動
- Management Console の設定
- データベースの設定
- ライセンス
- データの暗号化
- ロールと権限
- 高度な設定

Management Console の開始

最初に Management Console を起動します。

- Windows のスタート メニューから Management Console プログラムを選択します。
- コマンド ラインから Management Console 実行ファイルを実行します。

また、Windows サービスとして、またはコンテナ化された環境で実行するように設定することができます。

Management Console の構成

次の場所にある .properties ファイルを使用して Management Console を設定します。

[インストール_ディレクトリ]/WebApps/ManagementConsole

このフォルダには以下の設定ファイルが含まれています。

ファイル名	説明
application.properties	サーバー ポート、プラットフォーム DB 接続、SSL、および基本的なログ記録などの主要なアプリケーション設定。「設定プロパティ」を参照してください。
custom-roles.properties	ロールと権限の設定。「ロールと権限」を参照してください。
hazelcast.yml	高可用性を有効にするための Hazelcast の設定。「高可用性」を参照してください。
ldap.properties	LDAP の設定。「LDAP 認証」を参照してください。
log4j2.properties	高度なログ設定。「ロギングの設定」を参照してください。
saml.properties	SAML の設定。「SAML シングル サインオンの統合」を参照してください。

設定プロパティ

Management Console の設定ファイル application.properties には、次のプロパティが含まれています (わかりやすくするためにここではグループ化しています)。必要に応じて、このプロパティの設定を行います。たとえば、ポート 8080 で Management Console を実行するには、server.port=8080 と設定します。

コンテナ化された環境では、環境変数を使用してパラメータを設定します。「コンテナベースの展開」を参照してください。

時間ベースの設定を含むプロパティには、持続時間形式の数値を入力することができます。次のような接尾辞付きの簡単な表記を使用します: 秒は s、分は m、時間は h、日は d。

認証

プロパティ	説明
mc.auth.brute-force-protection.enabled	総当り攻撃に対する保護を有効にします。 デフォルト値: true
mc.auth.brute-force-protection.lockout-duration	ログイン失敗回数を超えた後のアカウント ロックアウトの期間。 デフォルト値: 5m

プロパティ	説明
mc.auth.brute-force-protection.max-attempts	アカウントがロックされるまでに許可されるログイン失敗回数。 デフォルト値: 3
mc.auth.shared-secrets.<service-name>	選択したサービスのサービス認証に使用される共有シークレット。 許可されるサービス名: das、dts、kapplets、RFS、roboserver、synchronizer、tota 設定されていない場合は、共有シークレットが自動的に生成されます。
mc.auth.superuser.create	スーパーユーザー アカウントが存在しない場合の自動作成を有効にします。 デフォルト値: true
mc.auth.superuser.name	自動的に作成されたスーパーユーザーのユーザー名。 デフォルト値: admin
mc.auth.superuser.password	自動生成されたスーパーユーザー用の初期パスワード。 デフォルト値: admin

タスク記録のクリーンアップ

プロパティ	説明
mc.clean-task-recordings.enabled	データベース内のタスク記録の自動クリーンアップを有効にします。 デフォルト値: true
mc.clean-task-recordings.duration-kept	クリーンアップ前にタスク記録がデータベースに保持される期間。 デフォルト値: 7d

データベース ドライバー

プロパティ	説明
mc.database-drivers.upload-from	データベース ドライバーのアップロードを許可されたソース。 可能な値: any_host、localhost、none。 Docker でユーザーによる JDBC データベース ドライバーのアップロードを許可するには、ANY_HOST に設定します。 デフォルト値: localhost

オフライン ドキュメント

プロパティ	説明
mc.help.offline-base-url	オフライン ドキュメントを含むフォルダへのパス。

高可用性

プロパティ	説明
mc.high-availability.enabled	高可用性モードを有効にします。 デフォルト値: false

KeyStore の設定

プロパティ	説明
mc.key-store.current.alias	現在の Java KeyStore ファイル内の証明書を参照するために使用されるエイリアス。 デフォルト値: mc
mc.key-store.current.location	機密データを暗号化するために使用された証明書を安全に格納し、その後データベースに保存する、現在の Java KeyStore ファイルへのパス。パスをファイル システムの位置として指定します。例: file:/c:/certs/mc.p12。 デフォルト値: classpath:mc.p12
mc.key-store.current.password	現在の Java KeyStore ファイルにアクセスしてロックを解除するためのパスワード。 デフォルト値: changeit
mc.key-store.previous.alias	以前の Java KeyStore ファイル内の証明書を参照するために使用されるエイリアス。
mc.key-store.previous.location	以前の Java KeyStore ファイルへのパス。 パスをファイル システムの位置として指定します。例: file:/c:/certs/mc.p12。
mc.key-store.previous.password	以前の Java KeyStore ファイルのパスワード。

ライセンス

プロパティ	説明
mc.license.server.api-key	Tungsten TotalAgility ライセンス API キー。
mc.license.server.max-design-studio-seats	Management Console が使用できる Design Studio シートの最大数。
mc.license.server.max-users	Management Console が処理することができるユーザーの最大数。
mc.license.server.non-production-cre-reserved	予約する非本番 CRE ライセンスの数。
mc.license.server.production-cre-reserved	Management Console によって起動時に割り当てられる本番 CRE ライセンスの数。
mc.license.server.total-agility-server-url	ライセンス サーバーとして機能する Tungsten TotalAgility インストールの URL。

プロパティ	説明
mc.license.static.company	ライセンスに定義されている会社名。
mc.license.static.non-production-key	非本番ライセンス キー。
mc.license.static.production-key	本番ライセンス キー。
mc.license.type	ライセンス タイプ。 可能な値: none、static、server。mc.license.type=server である場合は、mc.license.server という接頭辞が付いたプロパティが適用されます。 mc.license.type=static である場合は、mc.license.static というプレフィックスが付いたプロパティが適用されます。

設定されていない場合は、ユーザー インターフェイスでライセンス プロパティを設定することができます。

RoboServer クラスタ

プロパティ	説明
mc.robo-servers.clusters[N].connection-type	クラスタの接続タイプ。 可能な値: as_service、as_client。非推奨: as_service_ssl
mc.robo-servers.clusters[N].license-distribution	動的ライセンスの分配モードを有効にします。
mc.robo-servers.clusters[N].license-units	クラスタに割り当てられたライセンス ユニット。
mc.robo-servers.clusters[N].name	クラスタ名。
mc.robo-servers.clusters[N].production	クラスタ ライセンス タイプが「本番」であるかどうかを指定します。
mc.robo-servers.clusters[N].proxies[K].excluded-hosts	クラスタ プロキシの除外ホストのカンマ区切りのリスト。
mc.robo-servers.clusters[N].proxies[K].host	クラスタ プロキシ サーバーのホスト名。
mc.robo-servers.clusters[N].proxies[K].password	認証が必要な場合のクラスタ プロキシ パスワード。
mc.robo-servers.clusters[N].proxies[K].port	クラスタ プロキシ サーバーのポート。
mc.robo-servers.clusters[N].proxies[K].username	認証が必要な場合のクラスタ プロキシ ユーザー名。
mc.robo-servers.clusters[N].threshold-version	閾値の RoboServer バージョン。

プロパティ	説明
mc.robo-servers.robot-distribution-strategy	RoboServer スケーリング モード。 可能な値: load_balanced、scalable。 デフォルト値: load_balanced
mc.robo-servers.use-preset-clusters	Management Console の起動時にクラスタが存在しない場合は、mc.robo-servers.clusters プロパティを指定して、事前設定されたクラスタの作成を有効にします。 デフォルト値: false

N はクラスタ インデックス、K はプロキシ インデックスで、どちらも 0 から始まります。

Analytics データベース

プロパティ	説明
mc.settings.analytics-database.enabled	分析データベースの使用を有効にします。
mc.settings.analytics-database.host	分析データベース サーバーのホスト名。host:port という形式でポートを含めることができます (例: log-db-service:3307)。
mc.settings.analytics-database.keep-logs-days	統計を保持する日数。古いデータは、クリーンアップ設定に従って毎日削除されます。
mc.settings.analytics-database.password	分析データベースのパスワード。
mc.settings.analytics-database.schema	分析データベースのスキーマまたはカタログの名前。
mc.settings.analytics-database.type	分析データベースのデータベース タイプ。
mc.settings.analytics-database.username	分析データベースのアカウント名。

ログ データベース

プロパティ	説明
mc.settings.log-database.enabled	ログ データベースの使用を可能にします。
mc.settings.log-database.host	ログ データベース サーバーのホスト名。host:port という形式でポートを含めることができます (例: log-db-service:3307)。
mc.settings.log-database.keep-logs-days	ロボットとスケジュールの統計を保持する日数。古いデータは、クリーンアップ設定に従って毎日削除されます。
mc.settings.log-database.max-message-count-in-robot-run	ロボットの 1 回の実行での最大メッセージ数。この閾値を超えると、その実行のログ記録は停止します。
mc.settings.log-database.password	ログ データベースのパスワード。
mc.settings.log-database.schema	ログ データベースのスキーマまたはカタログの名前。
mc.settings.log-database.type	ログ データベースのデータベース タイプ。

プロパティ	説明
mc.settings.log-database.username	ログ データベースのアカウント名。

ベース URL

プロパティ	説明
mc.settings.base-url	Management Console ベース URL。

Desktop Automation サービス

プロパティ	説明
mc.settings.das-ping-interval	Desktop Automation サービスが Management Console に ping を送信する間隔 (ミリ秒)。

パスワード回復

プロパティ	説明
mc.settings.password-recovery-enabled	ユーザーがパスワードを紛失した場合またはパスワードを忘れた場合に復元できるように、電子メールによる通知の設定を有効にします。

プロキシ

プロパティ	説明
mc.settings.proxy.enabled	プロキシ サーバーの使用を有効にします。有効化されていない場合は、直接接続が使用されます。
mc.settings.proxy.host	プロキシ サーバーのホスト名。
mc.settings.proxy.password	認証が必要な場合のプロキシ パスワード。
mc.settings.proxy.port	プロキシ サーバーのポート。
mc.settings.proxy.username	認証が必要な場合のプロキシ ユーザー名。

SMTP

プロパティ	説明
mc.settings.smtp.encryption	SMTP 暗号化。 可能な値: none、TLS、smtps
mc.settings.smtp.host	SMTP サーバーのホスト名。
mc.settings.smtp.notification-from	通知の送信元の電子メール アドレス。
mc.settings.smtp.password	認証が必要な場合の SMTP サーバーのパスワード。
mc.settings.smtp.port	SMTP サーバーのポート。
mc.settings.smtp.username	認証が必要な場合の SMTP サーバーのユーザー名。

ロボット ファイル システム

プロパティ	説明
`mc.settings.rfs.enabled`	ロボット ファイル システム サーバーの使用を有効にします。
`mc.settings.rfs.url`	ロボット ファイル システム サーバーの URL。

設定されていない場合は、すべての `mc.settings` プロパティをユーザー インターフェイスで設定することができます。

テレメトリ

プロパティ	説明
`mc.telemetry.enabled`	テレメトリを有効にします。 デフォルト値: `true`

Vault

プロパティ	説明
`mc.vault.built-in-store.enabled`	組み込みの Vault ストアを有効にします。 デフォルト値: `true`
`mc.vault.cyber-ark.certificate-path`	CyberArk Central Credential Provider の TLS サーバー証明書へのパス。
`mc.vault.cyber-ark.enabled`	Vault 内の CyberArk ストアを有効にします。 デフォルト値: `false`
`mc.vault.cyber-ark.iis-application-name`	IIS で定義された CyberArk Central Credentials Provider のアプリケーション名。
`mc.vault.cyber-ark.port`	CyberArk Central Credentials Provider のホストのポート番号。
`mc.vault.cyber-ark.url`	CyberArk Central Credentials Provider のホストの URL。

設定されていない場合は、`mc.vault.cyber-ark` プロパティ (`enabled` 以外) をユーザー インターフェイスで設定することができます。

Spring Boot のプロパティ

Management Console は Spring Boot 上に構築されています。Management Console アプリケーションのプロパティとともに、次の組み込み Spring Boot プロパティを変更することもできます。

プロパティ	説明
`server.port`	サーバーの HTTP ポート。 デフォルト値: `50080`

プロパティ	説明
`server.ssl.*`	HTTPS 経由で Management Console をホストするためのプロパティのグループ。 詳細については、「HTTPS 上の Web ベースのコンポーネントのホスティング」を参照してください。
`spring.datasource.*`	データベースへの接続を設定するためのプロパティのグループ。

データベースの設定

Management Console は、次の 4 つのカテゴリに分類される複数のデータベース テーブルを使用します。

カテゴリー	説明
プラットフォーム テーブル	アップロードされたロボットやスケジュール情報などの Management Console データを保存します。
ログ テーブル	実行に関連するログを保存します。
分析テーブル	実行分析データを保存します。
収集テーブル	ロボットによって収集されたデータを保存します。RoboServer からアクセスできる必要があります。

プラットフォーム テーブルはプラットフォーム データベース内に存在します。Management Console を起動する前にプロビジョニングおよび設定を行う必要があります。ログ記録および収集データベースの設定は実行時に実行できます。詳細については、『Tungsten RPA のヘルプ』を参照してください。

デフォルトでは、Management Console は非本番環境のプラットフォーム データベースとして H2 データベースを使用します。このデータベースはテストおよび開発のみを目的としています。すべての H2 データベース ファイルは、以下の場所にローカルで保存されています。

- Windows: `C:\Users\[ユーザー]\AppData\Local\Tungsten RPA\[バージョン ]\Data`
- Linux: `/home/[ユーザー]/.Tungsten RPA/[バージョン ]/Data`

デフォルトのフォルダを変更するには、データベース接続の設定を変更してください。

本番環境では、Management Console を本番レベルのデータベースに接続するように設定します。

- データベースに関する考慮事項
- データベースの接続設定
- データベース ドライバー
- データベース アクセスと初期化

データベースに関する考慮事項

Management Console には、データベースへの低遅延ネットワーク パスが設定されている必要があります。最高のパフォーマンスと信頼性を得るには、Management Console とそのプラットフォーム データベースを同じリージョン内の、可能であれば同じ可用性ゾーンまたはデータ センターに展開します。それらが物理的に離れた場所に配置されている場合、ネットワークの待ち時間が増加し、小さなクエリが頻繁に実行されたときに遅延やタイムアウトが発生する可能性があります。

サポートされるデータベースのバージョンはリリース間で異なる場合があります。サポートされているデータベースとバージョンのリストについては、『Tungsten RPA 技術仕様』を参照してください。

次の要件に基づいて、Management Console テーブル専用のデータベースを作成することをお勧めします。

- Æ、ß、Ë などの非 ASCII 文字を処理するには、Unicode のサポートが必要です。
- 新しいデータベースを作成するときは、大文字と小文字を区別しない照合を使用してください。
- 大文字と小文字を区別する照合を使用したバックアップから復元を行う場合は、復元エラーや予期しない結果を回避するために、ターゲット データベースでも大文字と小文字を区別する照合を使用する必要があります。

⚠ 常に製品ドキュメントに従って Tungsten RPA データベースを作成および管理してください。ソフトウェアが動作しなくなる可能性があるため、Tungsten Automation に問い合わせることなくデータベースを変更またはカスタマイズしないようにしてください。

データベース サーバーは、Unicode と大文字と小文字の照合を異なる方法で処理します。次の表に、サポートされているデータベース システムに関する推奨事項を示します。

Unicode サポートと大文字と小文字の照合に関する推奨事項

データベース	推奨事項
MySQL	`utf8mb4_bin` 照合を使用してデータベースを作成します。
Oracle	Unicode には `NVARCHAR2` 型と `NCLOB` 型が使用されます。
Microsoft SQL Server	Management Console の場合は、`Latin1_General_100_BIN2` 照合を使用してデータベースを作成します。 Kapplets の場合は、`Latin1_General_100_BIN2_UTF8` 照合を使用してデータベースを作成します。 Unicode には `NVARCHAR` 型と `NTEXT` 型が使用されます。
PostgreSQL	`UTF-8` エンコーディングを使用してデータベースを作成します。

データベースの接続設定

Management Console をプラットフォーム データベースに接続するには、次の場所にある `application.properties` ファイルで接続設定を指定します。

`[インストール_ディレクトリ]/WebApps/ManagementConsole`

- `spring.datasource.url` - データベースへの JDBC 接続 URL (ドライバー固有の形式)。
- `spring.datasource.driver-class-name` - 完全修飾された JDBC ドライバー クラス名。
- `spring.datasource.username` - データベースのユーザー名。
- `spring.datasource.password` - データベースのパスワード。

データ ソースの URL と資格情報に従ってこれらの設定を更新します。環境に応じて、完全なデータベース設定に追加の設定が必要になる場合があります。使用している特定のデータベース タイプとバージョンについては、ドキュメントを参照してください。

例: MySQL データベース

```
spring.datasource.url=jdbc:mysql://database-service:3306/database-name?
autoReconnect=true&rewriteBatchedStatements=true

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.datasource.username=username

spring.datasource.password=password
```

例: Oracle データベース

```
spring.datasource.url=jdbc:oracle:thin:@database-service:1521:database-name

spring.datasource.driver-class-name=oracle.jdbc.driver.OracleDriver

spring.datasource.username=username

spring.datasource.password=password
```

例: PostgreSQL データベース

```
spring.datasource.url=jdbc:postgresql://database-service:5432/database-name?
currentSchema=schema-name

spring.datasource.driver-class-name=org.postgresql.Driver

spring.datasource.username=username

spring.datasource.password=password
```

例: SQL Server データベース

```
spring.datasource.url=jdbc:sqlserver://database-
service:1433;databaseName=database-name

spring.datasource.driver-class-
name=com.microsoft.sqlserver.jdbc.SQLServerDriver

spring.datasource.username=username

spring.datasource.password=password
```

例: SQL Server データベースと Windows 認証

統合セキュリティを使用している場合、Microsoft SQL Server JDBC ドライバーは、Management Console を実行する Windows アカウントを使用して認証します。ユーザー名とパスワードは使用されません。これを有効にするには、データ ソース URL に `integratedSecurity=true` を追加し、`spring.datasource.username` または `spring.datasource.password` を設定しないようにします。

```
spring.datasource.url=jdbc:sqlserver://database-
service:1433;databaseName=database-name;integratedSecurity=true

spring.datasource.driver-class-
name=com.microsoft.sqlserver.jdbc.SQLServerDriver
```

データベース ドライバー

プラットフォーム データベースに接続するために、Management Console には適切な JDBC ドライバーが必要です。展開タイプに応じて、次のようにドライバーを指定します。

- 非コンテナ: データベース ドライバーの JAR ファイルを次の場所に配置します。

  `[インストール_ディレクトリ]/lib/jdbc`
- コンテナ ベース: 環境変数 `JDBC_DRIVER_URL` をドライバー JAR の直接ダウンロード URL に設定します。

SQL Server Windows 認証 (統合セキュリティ) の場合は、ネイティブ認証 DLL (`sqljdbc_auth.dll` など) を `[インストール_ディレクトリ]\nativelib` に配置します。

Tungsten RPA ドキュメントで特に指定されていない場合は、プラットフォーム データベースと互換性のある最新の安定したドライバー バージョンを使用します。サポートされているバージョンについては、データベース ベンダーの公式ドキュメントを参照してください。

データベース アクセスと初期化

通常の操作では、Management Console が使用するデータベース ユーザー アカウントには、少なくとも SELECT、INSERT、UPDATE、DELETE、および CREATE TEMPORARY TABLES 権限が割り当てられている必要があります。

起動時に、Management Console は必要なテーブルが存在するかどうかを確認し、存在しない場合は作成します。この自動初期化用に、データベース ユーザーには CREATE TABLE、ALTER TABLE、および (Oracle の場合) CREATE SEQUENCE 権限が割り当てられている必要もあります。自動初期化を 1 度だけ使用する場合は、これらの昇格された権限を取り消して、最小限のランタイム権限のみを残すことができます。

または、次の場所にある SQL スクリプトを使用してテーブルを手動で初期化します。

`[インストール_ディレクトリ]/documentation/sql`

このディレクトリには、Management Console 用のスクリプトを含む次のサブディレクトリが含まれています。データベース エンジンに適切なスクリプトを実行します。

- `mc` - コア プラットフォーム テーブルを作成および削除するためのスクリプト。
- `Quartz` - Quartz スケジューリング フレームワーク用のスクリプト。手動で初期化を実行する場合は、これらのスクリプトを作成する必要があります。
- `logdb` - ログ テーブルを作成および削除するためのスクリプト。
- `statistics` - 分析テーブルを作成および削除するためのスクリプト。

ライセンス

Tungsten RPA を使用するには、有効なライセンスを取得する必要があります。ライセンス情報は Management Console 内で排他的に管理されます。他のすべての RPA コンポーネントは、ライセンスの詳細を取得するために Management Console に接続します。

Tungsten RPA のライセンスは、永続的または年単位で利用することができます。ライセンスは、対象のハードウェア環境や物理 CPU には依存しません。これにより、クラウドベース、オンプレミス、仮想化、または物理環境に柔軟に展開できるようになります。

- ライセンス モデル
- ライセンス キーのタイプ
- ライセンスの設定

Tungsten RPA のライセンスの詳細については、『Tungsten RPA のヘルプ』および Knowledge Portal を参照してください。

ライセンス モデル

Tungsten RPA は次の 2 つのライセンス モデルをサポートしています。

- 同時ロボット実行 (CRE) - 同時に実行できるロボットの数に基づいたライセンス。
- 計算ユニット (KCU) - RoboServer が 1 秒あたりに実行できる操作の数に基づくライセンス。

同時ロボット実行 (CRE)

同時ロボット実行 (CRE) ライセンスは、Tungsten RPA ロボット ライセンスに基づいています。ロボット ライセンスによって、RoboServer 上で同時に実行できるロボットの数が決定されます。たとえば、組織に 5 台のロボットのライセンスがある場合は、最大 5 台のロボットを同時に実行することができます。

設計や本番環境に展開することができるロボットの数に制限はありません。ロボットは要件に応じて、シンプルな構成または複雑な構成になります。たとえば、ロボットには、データの入力と取得、データの変換、Excel ファイルへの書き込み、および RESTful サービス API の呼び出しなどのワークフロー アクションを実行する任意の数のステップが含まれる場合があります。

計算ユニット (KCU)

計算ユニット (KCU) は、RoboServer が 1 秒あたりに実行できる操作 (ステップ) の数を測定します。この指標は物理的なサーバーの容量には依存しません。ステップとは、Web ページの読み込み、データベースへの書き込み、またはデータ要素の変換といった RoboServer が実行する最小のアクションです。

1 KCU とは 1 秒あたりのポイントが 5,000 KCU であることを表します。それぞれのタイプのステップでは、複雑さに応じて消費される KCU ポイントの数が異なります。主なステップ グループは以下のとおりです。

- I/O ステップと JavaScript ステップ: コストとして 10,000 KCU ポイントが消費されます。例: 4 KCU で 1 秒あたり 2 ページの読み込み。
- I/O ステップまたは JavaScript ステップ(両方ではない場合): コストとして 1,000 KCU ポイントが消費されます。例: 4 つの KCU で 1 秒あたり 20 回の REST Web サービス呼び出しステップ。
- 抽出ステップと変換ステップ: コストとして 1 KCU ポイントが消費されます。例: 4 KCU で 1 秒あたり 40,000 回の抽出または割り当てステップの実行。
- ループ ロボット ステップ: コストとして 1 回の反復で 1 KCU ポイントが消費されます。
- 10,000 KCU ポイントを消費する、ロボットのアプリケーション ステップ ([参照]、[Windows]、[Excel]、[ターミナル]、[Document Transformation]、[PDF]、および [電子メール])、[開く] ステップ (非推奨)、および [カスタム アクション] ステップ。
- Desktop Automation サービスに接続するロボットは起動時に 10,000 KCU ポイントを消費します。

完全な KCU のリストを見つけるには、Design Studio で [ヘルプ] > [KCU 情報を表示] を選択します。

信頼性の高いパフォーマンスを実現するには、システムに十分な CPU パワーがあることを確認し、ソース サーバーからの応答時間が適切な程度まで短くなるようにします。ロボットを実行した後に、Design Studio デバッガーの概要でロボットが使用する KCU の合計数を確認してください。

ライセンス キーのタイプ

使用可能なライセンス キーのタイプには次の 2 つがあります。

- **本番キー:** Tungsten RPA システムの本番使用を許可します。
- **非本番キー:** Tungsten RPA システムの非本番使用 (テストやステージングなど) を許可します。

各ライセンス キー タイプにより、使用可能な RPA 機能と、Management Console で設定された RoboServer クラスタで実行できるロボットの数が定義されます。

本番キーと非本番キーがある場合は、それらのキーを同じ Management Console インスタンスにインストールすることができます。または、キーごとに個別のシステムを設定することもできます。どちらの場合も、それぞれ本番環境および非本番環境として指定された、少なくとも 2 つの異なる RoboServer クラスタを設定する必要があります。これらのクラスタには、ライセンス内の CRE または KCU を割り当てることができます。

ライセンスの設定

Management Console のライセンスを設定するには、次のいずれかの方法を選択します。

- インストールに Tungsten TotalAgility ライセンス サーバーが含まれている場合は、Tungsten TotalAgility ライセンス サーバーを介した統合ライセンスを使用する。
- Management Console にライセンス キーを直接入力する。

Management Console では、ライセンスの方法を切り替えることができます。たとえば、ライセンス サーバーで問題が発生した場合、ライセンス キーに切り替えると、問題が解決されるまでユーザーは作業を続行することができます。

次のいずれかの方法でライセンス情報を提供します。

- スーパーユーザーとして、有効なライセンスなしで Management Console を起動し、表示されるメッセージに従ってログイン時にライセンス情報を入力します。
- スーパーユーザーとして、Management Console ユーザー インターフェイスの [管理] > [ライセンス] でライセンス情報を更新します。詳細な手順については、『Tungsten RPA のヘルプ』の「ライセンス」セクションを参照してください。
- `mc.license.server.*` および `mc.license.static.*` アプリケーション プロパティを使用してライセンス データを事前設定します。この方法では、完全なライセンス設定または部分的なライセンス設定を実行できます。詳細については、『Management Console の設定』を参照してください。

複数の Management Console インスタンスが個別のテナントとして動作している場合 (それぞれのインスタンスが独自のデータベースとユーザー ベースを持つ場合)、各インスタンスは固有のライセンス データを使用する必要があります。高可用性クラスタの一部であるすべての Management Console インスタンスは、同じライセンス データを使用する必要があります。

データの暗号化

Management Console は証明書ベースの暗号化 (公開鍵と秘密鍵) を使用して、パスワード、シークレット、およびアクセス トークンなどの機密データを安全に保存します。以前のバージョンからデータをインポートすると、新しい証明書ベースのアルゴリズムを使用してパスワードが自動的に再暗号化されます。

証明書とその証明書に一致する秘密鍵は Java KeyStore に保存されます。Management Console には、証明書と秘密鍵を含むデフォルトの KeyStore が含まれています。デフォルトの KeyStore はすべての顧客に対して同様であるため、独自の KeyStore を作成することをお勧めします。

次のトピックでは、独自の KeyStore を作成してアップグレードする方法について説明します。

- KeyStore の作成
- KeyStore のアップグレード

KeyStore の作成

Management Console がすでに実行されている場合は、証明書をアップグレードする必要があります。KeyStore は PKCS12 形式である必要があり、Java SDK に含まれている Keytool アプリケーションを使用して作成することができます。

次のコマンドにより、3650 日間 (10 年間) 有効な証明書を持つ新しい PKCS12 キーストアを作成します。

```
keytool -genkey -alias mc -keyalg RSA -validity 3650 -keystore mc.p12 -storetype pkcs12
```

このコマンドは、パスワードと X.509 秘密キーに保存されている情報の入力を要求します。現在のディレクトリに `mc.p12` (`-keystore` 引数の値) という名前のファイルを作成します。オプション `-validity 3650` により、証明書の有効期間を 10 年に設定します。

証明機関 (CA) によって発行された証明書を使用しないようにしてください。PKCS12 ファイルには秘密鍵および公開証明書が含まれており、秘密鍵のパスワードはアプリケーション設定に平文で表示されます。

Management Console が新しい証明書を使用するようにするには、次の場所にある `application.properties` ファイルで次のプロパティを指定します。

`[インストール_ディレクトリ]/WebApps/ManagementConsole`

- `mc.key-store.current.location` - ファイル システムの位置として指定された KeyStore ファイルへのパス。例: `file:/c:/certs/mc.p12`。
- `mc.key-store.current.password` - KeyStore ファイルにアクセスしてロックを解除するために使用されるパスワード。
- `mc.key-store.current.alias` - KeyStore ファイル内の証明書を参照するエイリアス。

KeyStore のアップグレード

Management Console を初めて起動すると、KeyStore の秘密鍵を使用してチェックサムが作成されます。このチェックサムにより、KeyStore が置き換えられたことを検知し、提供された証明書を使用してパスワードの復号が可能であることを確認することができます。

独自の KeyStore をインストールする前に Management Console をすでに起動している場合は、パスワード変換を実行するように Management Console を設定する必要があります。

KeyStore をアップグレードするには、現在の KeyStore ファイルを新しい場所にコピーし、`application.properties` ファイルで次のプロパティを指定して、以前の KeyStore を参照するパスワード コンバータを作成します。

- `mc.key-store.previous.location` - ファイル システムの場所として指定される、以前の KeyStore ファイルへのパス。例: `file:/c:/certs/previous-mc.p12`。
- `mc.key-store.current.password` - 以前の KeyStore ファイルにアクセスしてロックを解除するために使用されるパスワード。
- `mc.key-store.previous.alias` - 以前の KeyStore ファイル内の証明書を参照するエイリアス。

この設定により、パスワード コンバータは以前の証明書を使用して、既存の暗号化されたデータとチェックサムを復号できます。新しい秘密鍵 (以前に構成したもの) でデータが再暗号化され、新しいチェックサムが生成されます。

変換は、次に Management Console を起動したときに実行されます。データベース内のデータの量によっては、この変換に時間がかかる場合があります。変換が完了した後に、以前の KeyStore 設定を保持することができます。変換プロセスは、チェックサムとキーストアが同期していない状態となった場合にのみ再度実行されます。変換が完了すると、チェックサムは新しい KeyStore と一致するようになります。

ロールと権限

Management Console では、ユーザーの権限は、セキュリティ グループのメンバーシップを通じて割り当てられたロールに基づいています。組み込みロールは、設定可能なロールと固定ロールという 2 つのタイプに分けられます。さらに、特定の要件を満たす独自のロールを作成することもできます。

次のトピックでは、ロール、権限、およびユーザー管理の詳細について説明します。

- ロールの定義
- 権限のプロパティとアクション
- 組み込み RPA 管理者グループ
- 組み込み admin スーパーユーザー
- プロジェクト権限
- ユーザー ロールの概要については、『Tungsten RPA のヘルプ』の「ユーザーおよびグループ」を参照してください。

ロールの定義

固定ロールには、変更できない事前定義済みの権限があります。また、このロールにはサービス ロールも含まれますが、これは API アプリケーションでのみ使用され、ブラウザでの Management Console へのインタラクティブ ログインには使用されません。

設定可能なロールは、次の場所にある `.properties` ファイルで定義します。

`[インストール_ディレクトリ]/ManagementConsole.war/WEB-INF/classes/roles/`

これらのロールは次のプロパティを使用して定義し、ユーザー権限を調整するために変更することができます。

プロパティ名	必須	デフォルト値
mc.roles.<role-name>.name	はい	なし
mc.roles.<role-name>.permissions.<permission-name>	オプション	空

既存のロールの上書きやカスタム ロールの定義を行うするには、次の場所にある `custom-roles.properties` ファイルを変更します。

`[インストール_ディレクトリ]/WebApps/ManagementConsole/`

このファイルを使用して、任意の数の追加ロールを作成したり、要件に合わせてロールを設定したりすることができます。

- 権限に対して許可される複数のアクションを定義するには、それらのアクションをカンマで区切ります。リストに記載されていない権限は無効とみなされます。

  例:

  ```
  mc.roles.developer.name=Developer
  ```

```
mc.roles.developer.permissions.trigger-mappings=modify,view
mc.roles.developer.permissions.vault=viewOAuthClients,viewRobotAccess,viewSecrets
mc.roles.developer.permissions.schedules=modify,run,view
```

- このファイルの設定は、デフォルトのロール ディレクトリの設定よりも優先されます。
- 予期しない動作を回避するために、WAR ファイル内のロールは変更しないようにしてください。

権限のプロパティとアクション

このセクションには、Management Console のメニュー別に整理された、使用可能な権限のリストが含まれています。これらの権限とアクションの名前は `custom-roles.properties` ファイルで使用します。権限は Management Console の UI および API に適用されます。

コンテナ化された環境では、環境変数を使用してパラメータを設定します。「コンテナベースの展開」を参照してください。

Management Console オプション	権限プロパティ	アクション
ホーム	`message-of-day`	`view`
		`modify`
スケジュール	`schedules`	`view`
		`modify`
		`run` スケジュールの開始と終了に使用します。
[リポジトリ] > [ロボット]	`robots`	`view`
		`modify`
		`download`
		`run`
		`generateApiCode`
[リポジトリ] > [タイプ]	`types`	`view`
		`modify`
		`download`
		`executeSqlOnObjectDb`
[リポジトリ] > [スニペット]	`snippets`	`view`
		`modify`
		`download`
[リポジトリ] > [リソース]	`resources`	`view`
		`modify`
		`download`
		`modifyConnectors`

Management Console オプション	権限プロパティ	アクション
[リポジトリ] > [Vault]	vault	viewTargetSystems
		modifyTargetSystems
		viewSecrets
		modifySecrets
		extractSecrets Design Studio または RoboServer の対応するロボット ステップの実行時に、シークレット値を抽出できるようにします。
		viewOAuthClients
		modifyOAuthClients
		extractOAuthTokens Design Studio および RoboServer の対応するロボット ステップの実行時に、OAuth アクセス トークン値を抽出できるようにします。
		viewRobotAccess
		modifyRobotAccess
[リポジトリ] > [定数]	constants	modify
[リポジトリ] > [デバイス マッピング]	device-mappings	view
		modify
[リポジトリ] > [データベース マッピング]	database-mappings	view
		modify
[リポジトリ] > [ロボット ファイル システム]	rfs	view
		modify
		retrieve
[リポジトリ] > [電子メール トリガー]	email-triggers	view
		modify
[リポジトリ] > [トリガー マッピング]	trigger-mappings	view
		modify
データ ビュー	data-view	view
		modify
ログ ビュー	logs	viewScheduleRuns
		deleteScheduleRuns
		viewScheduleMessages
		deleteScheduleMessages
		viewRobotRuns

Management Console オプション	権限プロパティ	アクション
		deleteRobotRuns
		viewRobotMessages
		deleteRobotMessages
		viewRobotSummary
		viewRoboServerMessages
		deleteRoboServerMessages
		viewDasMessages
		deleteDasMessages
		viewTaskMessages
		deleteTaskMessages
[管理] > [タスク ビュー]	task-view	view
		stop
[管理] > [RoboServer]	clusters	viewClustersAndRoboServers
		viewClustersSettings
		modify
[管理] > [プロジェクト]	projects	view
		create
		update
		delete
[管理] > [デバイス]	devices	view
[管理] > [ユーザーおよびグループ]	users	view
		modify
設定	settings	viewGeneralSettings
		modifyGeneralSettings
		accessDesignStudioDesktopAutomations
		accessAllDesktopAutomations
		viewDesignStudioSettings
		modifyDesignStudioSettings
		modifyDatabaseTypes
		viewDatabaseDrivers
		modifyDatabaseDrivers
		retrieveProcessDiscoveryConfiguration
		viewTotalAgilityConfiguration
		modifyTotalAgilityConfiguration

Management Console オプション	権限プロパティ	アクション
		viewCyberArkConfiguration
		modifyCyberArkConfiguration
[設定] > [電子メール アカウント]	email-account	view
		modify
[ユーザー メニュー] > [ユーザー API キー]	user-account	modifyApiKeys
メニュー オプションには関連付けられておらず、グローバルに適用されます	general-ui	accessUi UI へのアクセスを許可します

組み込み RPA 管理者グループ

RPA 管理者グループのユーザーには、特別な admin スーパーユーザー権限を除く、すべてのプロジェクトに対するすべての権限があります。RPA 管理者は任意のプロジェクトで新しい管理者およびユーザーの作成を行います。ユーザーを管理者にするには、そのユーザーをこのグループに追加します。

- RPA 管理者グループは、内部ユーザーの管理が有効な場合に表示され、デフォルトでは空になっています。
- RPA 管理者には、次の Management Console [管理] 機能へのアクセス権はありません。
  - 高可用性ノード
  - サービス認証
  - バックアップ
  - ライセンス

組み込み admin スーパーユーザー

admin スーパーユーザー アカウントは、すべてのプロジェクトに無制限にアクセスすることができ、通常のプロジェクト権限をバイパスします。このアカウントは RPA 管理者グループのメンバーではないため、どのグループのメンバーにも属しません。

LDAP 統合セットアップでは、管理者グループは LDAP 設定の一部として定義されます。admin はログインを行い、開発者、プロジェクト管理者、RoboServer などのロールにマッピングする LDAP グループを定義できます。

admin スーパーユーザーは、内部ユーザー設定で初回の起動時に作成されます。このユーザーはログインして管理者、開発者、他のユーザーを作成することができます。

Management Console のバックアップを復元すると、デフォルトの admin スーパーユーザーがバックアップのスーパーユーザーに置き換えられます。復元した Management Console で指定された資格情報を使用します。

プロジェクト権限

新しいユーザーとグループを作成するには、**Management Console** > **[管理]** > **[ユーザーおよびグループ]** の順に移動します。セキュリティ モデルはロールベースです。ユーザーを作成した後に、手順の例に示したように、このユーザーを 1 つ以上のプロジェクトの特定のロールに関連付けられた 1 つ以上のグループに追加する必要があります。

同じセキュリティ グループに複数のロールを割り当てたり、同じロールを複数のセキュリティ グループに割り当てたりすることができます。ユーザーに複数のロールが割り当てられている場合、そのユーザーは少なくとも 1 つのロールで許可されているすべての操作を実行できます。Management Console で複数のプロジェクトを使用する場合は、グループをプロジェクト固有のロールに割り当てることで、異なるプロジェクトのユーザーを完全に分離することができます。

LDAP ユーザー アカウントを使用する方法については、『LDAP 認証』を参照してください。

例: 権限の割り当て

この例では、プロジェクト内の特定のロールに関連付けられたグループにユーザーを追加する方法を示します。

1. 「Developers」というグループを作成します。
   a. **[グループ]** タブで、プラス記号を選択します。
   [新しいグループを作成] ダイアログ ボックスが表示されます。
   b. **Developers** という名前を入力し、説明を入力して、**[OK]** を選択します。
   グループがテーブルに表示されます。
2. 「Dev」というユーザーを作成します。
   a. **[ユーザー]** タブで、プラス記号を選択します。
   [新しいユーザーを作成] ダイアログ ボックスが表示されます。
   b. ユーザー名 **Dev**、パスワード、氏名、および電子メールを入力します。グループを **Developers** に設定し、**[OK]** を選択します。
   「Dev」ユーザーがテーブルに表示されます。
3. 権限を割り当てます。
   a. 管理者としてログインし、**[管理]** > **[プロジェクト]** に移動します。
   b. デフォルト プロジェクトの ⋮ コンテキスト メニューで **[編集]** をクリックし、**[権限]** タブに移動します。
   [権限] タブには次の 2 つの列があります: **[プロジェクト ロール]** と **[セキュリティ グループ]**。プロジェクト ロールにより、ロボットの実行、スケジュールの変更、またはログの表示など、Management Console で使用可能な一連のアクションを定義します。プロジェクト内で、セキュリティ グループにプロジェクト ロールを割り当てます。そのセキュリティ グループ内のすべてのユーザーは、割り当てられたプロジェクト ロールによって定義された権限を継承します。
   c. このプロジェクトに権限を追加するには、プラス記号を選択します。

グリッドに新しい行が追加されます。

d. **[プロジェクト ロール]** で、**[開発者]** を選択します。

e. **[セキュリティ グループ]** で [**Developers**] (Dev ユーザーはこのグループに属します) を選択し、**[OK]** を選択します。
Developers グループのすべてのメンバーが、開発者ロールで許可されたアクションを実行できるようになりました。

f. 右上隅のメニュー ボタンを選択してログアウトし、Dev ユーザーとしてログインして、Management Console に権限がどのように反映されているかを確認します。

g. **[ログ ビュー]** に移動し、左側のペインで RoboServer のログインを選択します。
[削除] ボタンは無効になっています。ボタンのツールチップには、RoboServer のメッセージを削除する権限がないことが示されています。

高度な設定

高度な設定を使用して、システムの実行方法と通信方法を調整します。これらのオプションは、パフォーマンス、セキュリティ、および統合の詳細をカスタマイズする場合に役立ちます。

- ユーザーのオリジンと認証
- LDAP 認証
- SAML SSO 統合
- 高可用性
- ユーザー パスワードのハッシュ化
- テレメトリ
- RAM の割り当ての変更
- ログ記録の設定

ユーザーのオリジンと認証

Management Console にログインしたユーザーは、ユーザー名とオリジンによって識別されます。Management Console の **[ユーザーおよびグループ]** ページの [ユーザーのオリジン] フィールドには、次の表に示すように、ユーザーの作成方法に関する情報が表示されます。

ユーザーのオリジン	説明
`unknown`	ユーザーはバックアップの復元後に作成されました。
`internal`	ユーザーは [ユーザーおよびグループ] ページで手動で作成されました。
`saml`	ユーザーは SAML 経由でログインした後に作成されました。
`ldap#{ldapDirectoryIdentifier}`	ユーザーは LDAP 経由でログインした後に作成されました。

ユーザー認証を行う場合は以下の内容に注意してください。

- あるユーザーがログインしたときに、同じ名前を持ち、オリジンが `unknown` であるユーザーがすでに存在する場合、新たなユーザーが作成されることはなく、新しいログイン方法に基づいて既存のオリジンが変更されます。
- SAML や LDAP などの外部 ID プロバイダを使用していない場合は、Management Console の **[ユーザーおよびグループ]** ページで、選択したユーザーの **[内部オリジンを設定]** の下にある `unknown` というオリジンを `internal` に変更します。

LDAP 認証

Tungsten RPA は、Lightweight Directory Access Protocol (LDAP) を介した認証をサポートしています。

次のトピックでは、LDAP 統合、設定プロパティ、SSL 接続エラーの解決について詳しく説明します。

- LDAP 統合
- 設定プロパティ
- LDAPS 使用時の SSL 接続エラーの解決方法

⚠ CA Single Sign-On のサポートは、Tungsten RPA 2026.1 以降では廃止されます。この機能は今後のいずれかのリリースで削除される予定です。CA シングル サインオンを使用している場合は、2026.1 以降へのアップグレード前に LDAP または SAML などの代替手段に移行することをお勧めします。

LDAP 統合

LDAP 認証を有効にして設定するには、次の場所にある `ldap.properties` ファイルを変更します。

`[インストール_ディレクトリ]/WebApps/ManagementConsole/`

1. LDAP を有効化します。
   a. `mc.auth.ldap.enabled`を`true` に設定します。
   b. `mc.auth.ldap.directories` に少なくとも 1 つの LDAP ディレクトリを設定します。
2. 複数の LDAP ディレクトリを設定します。
   Tungsten RPA はマルチフォレスト LDAP 統合をサポートしており、複数の LDAP ディレクトリに同時に接続することができます。
   それぞれの LDAP ディレクトリは、次のプロパティ パターンを使用して設定します。
   `mc.auth.ldap.directories[N].*`
   ここで N はディレクトリのインデックスで、0 から始まります。
   - 単一のディレクトリの場合は、`mc.auth.ldap.directories[0].*` を使用します。
   - 2 つ目のディレクトリには、`mc.auth.ldap.directories[1].*` を使用します。
   - 追加のディレクトリのインデックスについては、[ ] 内の数を増やします。
3. Management Console アクセスを設定します。
   LDAP アカウントを使用して Management Console を管理するには、ユーザーが属する LDAP グループの 1 つを `mc.auth.ldap.directories[N].superuser-groups` に追加します。

このプロパティにリストされているグループのメンバーには、Management Console でのスーパーユーザー権限が付与されます。

アクセスを制限するには、スーパーユーザー専用の LDAP グループを作成します。

4. グループの可視性を設定します。

   Management Console でプロジェクト権限を割り当てると、使用可能なグループのリストが LDAP から入力されます。

   グループは、すべてのグループを取得するためのフィルタを構築する `mc.auth.ldap.directories[N].group-role-attribute` プロパティを使用して検索されます。

   すべての LDAP グループを表示する必要がない場合は、`mc.auth.ldap.directories[N].all-groups-filter` で独自のフィルタを定義してデフォルトの動作を上書きします。

   - `(cn=*)` - グループ名が `cn` 属性に含まれている場合に、すべてのグループ名を検索します (デフォルト)。
   - `(cn=E*)` - 文字「E」で始まるグループを検索します。

   フィルタは標準の LDAP クエリ構文を使用します。

   > グループ名は、リスト内のすべての LDAP サーバーで一意である必要があります。

   詳細については LDAP のドキュメントを参照してください。

5. [セキュアな LDAP]:

   Tungsten RPA は、Management Console での LDAPS (LDAP over SSL) をサポートしています。

   LDAPS を使用するには、`mc.auth.ldap.directories[N].server-url=ldaps://<hostname>:<port>` を設定します。

   デフォルトでは、LDAPS ポートは 636 です。

設定プロパティ

LDAP 設定ファイル `ldap.properties` には、次のようなプロパティが含まれています (わかりやすくするためにここではグループ化しています)。

LDAP を有効化

プロパティ	説明
`mc.auth.ldap.enabled`	LDAP を有効化するには、`true` に設定します。 デフォルト値: `false`

ディレクトリ識別とロールのマッピング

プロパティ	説明
mc.auth.ldap.directories[N].id	Management Console 内のユーザーのオリジン フィールドの一部として使用される LDAP ディレクトリ識別子。この名前は、LDAP ディレクトリごとに一意である必要があります。 注意: 複数の LDAP ディレクトリで同じ識別子が使用されている場合、Management Console は起動しません。 デフォルト値: 0
mc.auth.ldap.directories[N].superuser-groups	すべての機能にアクセスできる、Management Console 内のスーパーユーザーにマッピングされた LDAP グループのリスト。convert-to-upper-case を true にした場合は、大文字のグループ名が使用されます。 デフォルト値: SUPERUSERS
mc.auth.ldap.directories[N].rpa-administrator-groups	Management Console の RPA 管理者にマッピングされた LDAP グループのリスト。これらのグループのメンバーには、LDAP グループをロールにマッピングするなど、すべてのプロジェクトに対する (admin ユーザーの特別な権限以外の) すべての権限があります。convert-to-upper-case を true にした場合は、大文字のグループ名が使用されます。 デフォルト値: RPAADMINISTRATORS
mc.auth.ldap.directories[N].convert-to-upper-case	グループ名を大文字に変換するには、true に設定します。 デフォルト値: true
mc.auth.ldap.directories[N].all-groups-filter	プロジェクトの権限の作成時に表示されるグループを制御します。 デフォルト値: (cn=*)

接続設定

プロパティ	説明
mc.auth.ldap.directories[N].server-url	LDAP サーバーの URL で、ldap:// または ldaps:// プロトコルを使用します。 デフォルト値: ldap://localhost:389
mc.auth.ldap.directories[N].connect-timeout	LDAP サーバーへの接続のタイムアウト。秒の場合は接頭辞 s、分の場合は接頭辞 m を使用して指定します。 デフォルト値: 30s
mc.auth.ldap.directories[N].read-timeout	最初の接続が確立された後の LDAP サーバーからの応答のタイムアウト。秒の場合は接頭辞 s、分の場合は接頭辞 m を使用して指定します。 デフォルト値: 30s

認証アカウント

プロパティ	説明
mc.auth.ldap.directories[N].user-dn	LDAP にログインして他のユーザーを認証するために使用される識別名 (DN)。 デフォルト値: cn=admin,dc=example,dc=org
mc.auth.ldap.directories[N].password	user-dn アカウントのパスワード。 パスワードは平文で保存されるため、読み取り専用アクセス権が割り当てられたアカウントを使用してください。

検索設定

プロパティ	説明
mc.auth.ldap.directories[N].user-search-base	ユーザーが配置されている LDAP ツリー内のサブディレクトリ。 デフォルト値: ou=People,dc=example,dc=org
mc.auth.ldap.directories[N].user-search-filter	ユーザー名を検出するために適用するフィルタ。 デフォルト値: (uid={0})
mc.auth.ldap.directories[N].user-search-subtree	ユーザーが user-search-base のサブディレクトリに配置されている可能性がある場合は true に設定します。 デフォルト値: true
mc.auth.ldap.directories[N].group-search-base	LDAP ツリー内のグループが配置されているサブディレクトリ。 デフォルト値: ou=Groups,dc=example,dc=org
mc.auth.ldap.directories[N].group-search-filter	グループ内のユーザーを識別するために適用するフィルタ。 デフォルト値: (member={0})
mc.auth.ldap.directories[N].group-role-attribute	グループ名を保持する属性。 デフォルト値: cn
mc.auth.ldap.directories[N].group-search-subtree	グループが group-search-base のサブディレクトリに配置されている可能性がある場合は true に設定します。 デフォルト値: true

属性マッピング

プロパティ	説明
mc.auth.ldap.directories[N].full-name-attribute	ユーザーの氏名を取得するために使用する属性。 デフォルト値: displayName

プロパティ	説明
`mc.auth.ldap.directories[N].email-attribute`	ユーザーの電子メール アドレスを取得するために使用する属性。 デフォルト値: `mail`

紹介の処理

プロパティ	説明
`mc.auth.ldap.directories[N].referral`	LDAP ツリー内のサブノードへのリダイレクトを許可するには、`follow` に設定します。 デフォルト値: `follow`

LDAPS 使用時の SSL 接続エラーの解決

LDAPS の使用中に接続エラーが発生した場合は、次を確認してください。

- LDAPS では、LDAP サーバーによって提示された証明書が Tomcat を実行している Java によって信頼されている必要があります。アプリケーションが使用する Java キーストアにパブリック証明書をインポートします。
- JRE または JDK のインスタンスが複数ある場合など、証明書が正しいトラストストアにインポートされていることを確認してください。
- 正しいトラストストアが使用されていることを確認してください。`-Djavax.net.ssl.trustStore` が設定されている場合、デフォルトのトラスト ストアの位置をオーバーライドします。
- Exchange などのメール サーバーに接続する場合は、認証でプレーン テキストが許可されていることを確認してください。
- ターゲットサーバーが SSL を正しく提供するように設定されていることを確認します。SSL サーバー テスト ツールを使用します。
- ウイルス対策ツールに SSL および TLS をブロックする「SSL スキャン」があるかどうかを確認します。この機能を無効にするか、ターゲット アドレスに例外を設定します。

SAML シングル サインオンの統合

Management Console は、一元的な ID 管理のために SAML シングル サインオン (SSO) によるユーザー事前認証をサポートします。Management Console は、SAML 2.0 準拠の ID プロバイダ (IdP) との互換性を持つ標準の Spring Boot および OpenSAML ライブラリを使用します。

直接統合が確認されている IdP のリストについては、『Tungsten RPA 技術仕様』を参照してください。

SAML 統合には有効なライセンスが必要です。Management Console の起動前に必ずライセンスをインストールしてください。

SAML SSO を有効にするには、IdP および Management Console SAML の設定を行います。

- 外部 IdP の設定
- SAML の設定プロパティ

- SAML シングル ログアウトの有効化

外部 IdP の設定

IdP の設定手順は、特定のプロバイダに応じて異なります。設定手順については IdP の公式ドキュメントを参照してください。

Management Console の固有の設定を設定するには、次の手順を実行します。

1. Assertion Consumer Service (ACS) のエンドポイントを次の形式で Management Console の URL に設定します。`<MC URL>/login/saml2/sso/idp`

   例: `https://mc-service:8443/login/saml2/sso/idp`

   SAML 認証応答はこの URL に送信されます。
2. SAML メッセージの署名が必要な場合:
   a. 証明書と秘密鍵を生成します。
   b. 署名検証用に証明書を IdP にアップロードします。

SAML の設定プロパティ

次の場所にある `saml.properties` ファイルで SAML SSO を設定します。

`[インストール_ディレクトリ]/WebApps/ManagementConsole/`

Management Console で SAML SSO を有効にするには、次の手順を実行します。

1. `mc.auth.saml.enabled` を `true` に設定します。
2. `mc.auth.saml.*` プロパティを設定します。
3. `spring.security.saml2.relyingparty.registration.idp.*` プロパティを設定します。

SAML SSO の有効化

プロパティ	説明
`mc.auth.saml.enabled`	SAML SSO を有効にします。 デフォルト値: `false`

アサーション属性

プロパティ	説明
`mc.auth.saml.assertion-attributes.groups.names`	SAML 応答内の IdP からのグループ割り当て属性。 デフォルト値: `memberOf`
`mc.auth.saml.assertion-attributes.groups.separator`	グループ名の区切り文字。 デフォルト値: `;`
`mc.auth.saml.assertion-attributes.users.email`	SAML 応答内のユーザーの電子メール属性。 デフォルト値: `email`

プロパティ	説明
`mc.auth.saml.assertion-attributes.users.firstname`	SAML 応答内のユーザーの名属性。 デフォルト値: `firstname`
`mc.auth.saml.assertion-attributes.users.lastname`	SAML 応答内のユーザーの姓属性。 デフォルト値: `lastname`
`mc.auth.saml.assertion-attributes.users.regex`	指定した条件に一致しない名前を持つユーザーをフィルタリングするための正規表現。デフォルト値: `<regular expression allowing a wide variety of Unicode characters>`

グループ管理

プロパティ	説明
`mc.auth.saml.group-management.user-group-assignment`	ユーザー グループの割り当て戦略: `idp`: グループは IdP によって完全に決定されます。 `mc`: Management Console でグループを手動で割り当てることができます。 デフォルト値: `idp`
`mc.auth.saml.group-management.superusers`	フルアクセス権を持つグループ (スーパーユーザー)。 デフォルト値: `Admins`
`mc.auth.saml.group-management.rpa-administrators`	RPA 管理者ユーザー グループとして割り当てられたグループ。 デフォルト値: `RpaAdministrators`

ログイン動作

プロパティ	説明
`mc.auth.saml.landing-page`	`false` に設定した場合、ユーザーは自動的に IdP ログインにリダイレクトされます。 `true` に設定した場合、ユーザーには最初に SAML ログインが表示されます。 デフォルト値: `false`
`mc.auth.saml.single-logout`	`false` に設定した場合は、Management Console からログアウトするとセッションが終了し、確認メッセージが表示されます。 `true` に設定した場合は、Management Console からログアウトすると IdP からもログアウトされます。 デフォルト値: `false`

Spring セキュリティ IdP プロパティ

プロパティ	説明
`spring.security.saml2.relyingparty.registration.idp.entity-id`	IdP 用の Management Console の一意の識別子。ここでは Management Console のルート URL を使用できます (例: `https://mc-service:8443/`)
`spring.security.saml2.relyingparty.registration.idp.assertingparty.metadata-uri`	IdP メタデータ XML ファイルの場所。この値は、IdP 設定に登録されている SAML アプリケーションから取得します。
`spring.security.saml2.relyingparty.registration.idp.singlelogout.response-url`	SAML シングル ログアウト応答の相対 URL。この値を `{baseUrl}/logout/saml2/slo` に設定します
`spring.security.saml2.relyingparty.registration.idp.signing.credentials[0].private-key-location`	SAML リクエストに署名するための秘密鍵の場所。
`spring.security.saml2.relyingparty.registration.idp.signing.credentials[0].certificate-location`	SAML リクエストに署名するための秘密鍵に対応する X.509 証明書の場所。

SAML シングル ログアウトの有効化

Management Console は SAML シングル ログアウト (SLO) をサポートしており、ユーザーは 1 度のアクションで IdP からログアウトできます。SLO が有効化され設定されている場合、Management Console からログアウトすると IdP へのログアウト要求がトリガーされ、IdP は、ユーザーがアクティブな SSO セッションを持つすべてのアプリケーション (Management Console を含む) にわたってログアウト プロセスを調整します。

SAML シングル ログアウトを有効にするには、次の手順を実行します。

1. `mc.auth.saml.single-logout` を `true` に設定します。
2. `spring.security.saml2.relyingparty.registration.idp.singlelogout.response-url` を使用して SLO 応答 URL を設定します。
3. ログアウト サービスのリダイレクト バインディング URL として Management Console SLO エンドポイントを使用するように IdP を設定します。

高可用性

高可用性 (フェイルオーバー) により、ハードウェア障害、ソフトウェアの問題、またはネットワークの中断が発生しても、システムが機能し続けることが保証されます。高可用性が必要な場合は、複数の Management Console インスタンスをクラスタとして連携するように設定します。完全なフェールオーバーを実現するには、次のコンポーネントをクラスタ化する必要があります。

コンポーネント	説明
Management Console	同一の Management Console インスタンスが 2 つ以上必要です。
ロード バランサー	複数の Management Console インスタンス間でリクエストを分散するには、HTTP ロード バランサーが必要です。

コンポーネント	説明
クラスタ化されたプラットフォーム データベース	Management Console は、スケジュール、ロボット、およびその他のデータをプラットフォーム データベースに保存します。フェイルオーバー シナリオでは、単一障害点を回避するために、プラットフォーム データベースをクラスタ化された DBMS で実行する必要があります。

次のトピックでは、高可用性を有効にするための Management Console の設定方法、およびアプリケーションが適切にクラスタ化されていることを確認する方法について説明します。

- 高可用性を有効にするための Management Console の設定
- クラスタの状態の確認

高可用性を有効にするための Management Console の設定

高可用性を有効にするために複数の Management Console インスタンスを展開する場合、それらのインスタンスは同期されたクラスタとして動作する必要があります。この調整は、実行中のすべてのインスタンス間で共有されるデータ構造と通信を管理する Hazelcast によって提供されます。

たとえば、RoboServer 上でロボットを実行すると、特定の Management Console インスタンス内のスレッドでロボットのステータス メッセージが処理されます。クラスタ環境では、ユーザーが別の Management Console インスタンスから停止コマンドを送信する場合があります。Hazelcast はコマンドをすべてのインスタンスにブロードキャストし、ロボットを実行しているインスタンスがリクエストを受け取ってロボットを停止します。

Management Console の高可用性を有効にするには、次の手順を実行します。

1. `[インストール_ディレクトリ]/WebApps/ManagementConsole` に移動します。
2. `application.properties` ファイルで、次のプロパティを設定します: `mc.high-availability.enabled=true`。
3. `hazelcast.yml` ファイルで、高可用性クラスタにネットワーク設定を設定します。

   Hazelcast は、クラスタ メンバーの通信方法を制御するための柔軟なネットワーク設定プロパティを提供します。主なプロパティは以下のとおりです。

プロパティ	説明
`hazelcast.network.port`	Hazelcast メンバーが受信した接続に使用するポートまたはポート範囲。
`hazelcast.network.join`	Hazelcast メンバーがメンバーを互いに検出してクラスタを形成する方法を指定します。一般的なオプションは以下のとおりです。 • マルチキャスト: メンバーはマルチキャストを使用して互いに検出を行います。 • TCP/IP: 直接検出用の IP アドレスまたはホスト名のリストを使用してメンバーを設定します。マルチキャストが利用できない場合、またはマルチキャストが必要ではない場合に使用されます。

プロパティ	説明
`hazelcast.network.interfaces`	Hazelcast がバインドするネットワーク インターフェイスを制御します。デフォルトでは、Hazelcast はすべてのローカル インターフェイスにバインドしますが、セキュリティまたはネットワーク トポロジ上の理由から、これを特定のインターフェイスに制限することもできます。
`hazelcast.network.ssl`	メンバー間およびクライアントとメンバー間の通信で SSL/TLS 暗号化を有効にし、転送中のデータのセキュリティを強化します。

詳細や設定の例については、Hazelcast のドキュメントを参照してください。

4. クラスタ内のすべての Management Console インスタンスに同じ設定が適用されるようにします。
5. 最初の Management Console インスタンスを起動し、完全に起動するまで待ちます。設定されていない場合はライセンスを入力します。
6. 最初のインスタンスが実行中であることを確認した後に、残りの Management Console インスタンスを起動します。

   高可用性モードでは、すべての Management Console インスタンスが同じライセンス設定を使用するようにする必要があります。

- 他のすべてのサービス (RoboServer、Kapplets) は、ロード バランサーを介して Management Console にアクセスするように設定する必要があります。直接接続しないようにしてください。スティッキー セッションを有効にすると、セッションの永続性を維持する場合に役立ちます。
- クライアント接続モードで実行されている RoboServer は、高可用性モードではサポートされません。
- Docker で、`hazelcast.yml` ファイルをコンテナにマウントし、このファイルを指すように `SPRING_HAZELCAST_CONFIG` 環境変数を設定します。

クラスタの状態の確認

高可用性を設定した後に、Management Console クラスタが正しく実行されていることを確認します。

[Management Console] > **[管理]** > **[高可用性ノード]** にアクセスし、以下の点を確認してください。

- [インターフェイス] 列には、Hazelcast がクラスタ間の通信に使用している IP、またはホストとポートが表示されます。
- [接続先] 列には現在ユーザーが接続しているノードが表示されます。
- 接続しているサーバーをシャットダウンすると、ロード バランサーは自動的にユーザーを別のライブ インスタンスに転送します。
- 任意のノードのコンテキスト メニューから、デバッグ目的でスレッド ダンプをリクエストすることができます。

ユーザー パスワードのハッシュ化

Management Console では、Argon2 パスワード ハッシュ関数を使用してユーザー パスワードが保存されます。この実装は、ソルト長 16 バイト、ハッシュ長 32 バイト、並列度 1、メモリ コスト 2^{14}、および反復回数 2 というパラメータを持つデフォルトの Argon2 メカニズムに基づいています。

テレメトリ

Tungsten Automation はテレメトリ機能を使用して特定の指標を収集し、製品を分析および改善します。

Tungsten RPA では、テレメトリ機能は Management Console にのみ適用されます。テレメトリ機能はデフォルトで有効になっており、収集されたデータは 24 時間ごとに Tungsten Automation テレメトリ サービスに報告されます。

テレメトリ機能を無効にするには、`application.properties` ファイルで `mc.telemetry.enabled` を `false` に設定します。

RAM の割り当ての変更

インストール時に、Management Console はデフォルトで最大 8 GB という RAM の割り当てで設定されます。これは通常、一般的なワークロードには十分なサイズですが、多数のロボットを並行して実行する場合や大量のメモリを使用するロボットがある場合は、この制限を増やす必要があることがあります。

Management Console のメモリ割り当てを変更するには、次の手順を実行します。

1. `[インストール ディレクトリ]/bin/` に移動し、テキスト エディターで `ManagementConsole.conf` ファイルを開きます。
2. `wrapper.java.maxmemory` パラメーターを含む行を見つけます。
   行が存在しない場合は、追加してください。
3. 先頭の # を削除して行のコメントを解除し、その値を編集します。
   たとえば、Management Console が最大 16 GB の RAM を使用できるようにするには、次のように入力します: `wrapper.java.maxmemory=16384`。

ロギングの設定

ログの記録を調整して、問題のトラブルシューティングやシステムのパフォーマンスを監視します。

Management Console はログ記録に Log4j2 フレームワークを使用します。アプリケーション ログ ファイルはアプリケーション データ ディレクトリ内の [ログ] フォルダに保存されます。これらのフォルダの詳細については、『Tungsten RPA インストール ガイド』を参照してください。

- 基本設定
- 監査ログと高度な設定

基本設定

ログ記録に対する変更は、多くの場合、ログ記録レベルの調整のみとなります。次の場所にある `application.properties` ファイルにプロパティを追加します。

`[インストール_ディレクトリ]/WebApps/ManagementConsole`

たとえば、Quartz に対してデバッグ レベルのログ記録を有効にするには、次のプロパティを追加します: `logging.level.org.quartz=debug`

詳細については、Spring Boot のログ設定に関するドキュメントを参照してください。

監査ログと高度な設定

Management Console の監査ログには、API 呼び出しを含むシステム内のユーザー操作が記録されます。このログには、設定やデータベース コンテンツの変更が発生するアクティビティ、および認証イベントが記録されます。

監査情報は、ファイル、データベース、または Log4j2 でサポートされているその他のターゲットに記録することができます。

デフォルトでは、監査ログは INFO ログ レベルで有効になっており、標準出力およびメインの Management Console ログ ファイルに書き込まれます。

次のファイルで監査ログとその他の詳細プロパティを構成します。

`[インストール_ディレクトリ]\WebApps\ManagementConsole\log4j2.properties`

Docker で、外部の `log4j2.properties` ファイルをマウントし、`LOGGING_CONFIG` 環境変数を使用してそのパスを指定します。次に例を示します。`LOGGING_CONFIG=/ext/log4j2.properties`。

詳細については、Log4j2 のドキュメントを参照してください。

例: ファイルへのログ記録

「警告」ログ レベルでファイルに監査ログを設定するには、次の設定を使用します。

```
# Audit File Appender
appender.audit.type=File
appender.audit.name=AuditFile
appender.audit.fileName=logFilePath/logFileName.log
appender.audit.layout.type=PatternLayout
appender.audit.layout.pattern=%d{yyyy-MM-dd HH:mm:ss.SSS} %-5level %msg%n

# Audit Logger
logger.audit.name=auditLog
logger.audit.level=warn
logger.audit.appenderRef.audit.ref=AuditFile
logger.audit.additivity=false
```

第 3 章

RoboServer

RoboServer は、クライアント向けサービスとしてロボットを実行する Tungsten RPA のアプリケーションです。Design Studio で作成されたロボットを実行し、制御、ライセンス、およびスケジューリングのために Management Console に接続します。

RoboServer を起動すると、ロボットの実行要求などのクライアントからの要求を受け入れ、ロボットが実行中に抽出したオブジェクトなどの応答を送り返します。ロボットは、Management Console を使用して特定の時間に実行するようにスケジューリングするか、REST Web サービスを介した呼び出し、Java または .NET API を介した呼び出し、あるいは Kapplets からの呼び出しを行うことで起動できます。

RoboServer は Management Console と次の 3 つの方法で接続することができます。

- **クライアント接続**: RoboServer は Management Console への HTTP(S) 接続を作成し、指定されたクラスタに登録されます。この接続タイプは、「クライアント」タイプのクラスタでのみ使用してください。通常は、RPA がクラウド環境に展開されている場合にこの接続タイプを使用します。

  [クライアント接続] は高可用性モードではサポートされていません。

- **ソケット サービス**: RoboServer はサービスとして機能します。指定されたクラスタに RoboServer が登録された後に、Management Console はソケット接続を開始します。この接続タイプは、「サービス」タイプのクラスタでのみ使用してください。
- **ソケット SSL サービス**: (RPA 2026.1 では非推奨) RoboServer サービスとして機能します。指定されたクラスタに RoboServer が登録された後に、Management Console は暗号化されたソケット接続を開始します。この接続タイプは、「Service SSL」タイプのクラスタでのみ使用してください。

ロボットを実行するには、RoboServer を Management Console でアクティブ化する必要があります。

RoboServer は次の場合にアクティブになります。

- Management Console のクラスタに属している。
- クラスタに有効なライセンスがある。
- クラスタに十分な CRE または KCU が割り当てられている。

Management Console は、クラスタ上で設定された RoboServer からの設定も受信します。

RoboServer およびクラスタの管理の詳細については、『Tungsten RPA のヘルプ』の「Management Console」セクションを参照してください。

RoboServer を設定および管理するには、次のトピックを参照してください。

- RoboServer の起動

- RoboServer の設定
- RAM の割り当ての変更
- RoboServer のシャットダウン
- RoboServer サービスの起動のトラブルシューティング

RoboServer の開始

最初に RoboServer を起動します。

- Windows のスタート メニューから RoboServer プログラムを選択します。
- コマンド プロンプト ウィンドウを開き、Tungsten RPA インストール ディレクトリの `bin` フォルダに移動して、次のように入力します: RoboServer

  次の場所にある設定ファイルに必要なすべてのパラメータが指定されている場合、RoboServer が起動します。

  ```
  C:Users\[ユーザー]\AppData\Local\[バージョン]\Tungsten RPA\Configuration
  \roboserver.settings
  ```

  必要なパラメータが不足している場合は、使用方法の説明および使用可能なパラメータとともにエラー メッセージが表示されます。
- コマンド ラインに以下のパラメータを入力します。

  ```
  RoboServer [-client] [-s <service:params>] [-mcUrl <url>] [-ss <MC shared secret> [-
  cl <Cluster Name>] [-b <url>] [-p <port number>] [-sslPort <port number>] [-v]
  ```

  詳細については、「コマンド ラインの設定パラメータ」を参照してください。

また、Windows サービスとして、またはコンテナ化された環境で実行するように設定することができます。

RoboServer の構成

以下のトピックでは、RoboServer を設定する方法について説明します。

- RoboServer の自動登録
- RoboServer 設定
- コマンド ラインの設定パラメータ
- セキュリティ
- JMX サーバーの設定
- 本番構成

RoboServer の自動登録

すべての RoboServer は Management Console に自動的に登録されるようにする必要があります。この設定を行うには、次の手順を実行します。

1. RoboServer を開始します。
2. RoboServer 設定アプリケーションを開きます。
3. **[一般]** タブで、次のフィールドに入力します。**[Management Console URL]**、**[共有シークレット]**、および **[クラスタ]**。
   [OK] を選択して変更を保存します。次に、変更を有効にするために RoboServer を再起動します。

RoboServer 設定

RoboServer 設定アプリケーションを使用して RoboServer を設定するには、Windows のスタート メニューから RoboServer 設定を起動します。

設定アプリケーションのタブを使用して、以下の設定を行います。

- 一般: RoboServer 接続オプション、共有シークレットを含む Management Console 接続オプション、RoboServer ホスト設定、ライセンス ユニットの数、およびその他のオプション。
- セキュリティ: 権限などのセキュリティ設定。
- 証明書: TLS 証明書の使用。
- JMX サーバー: JMX サーバーの設定。

設定を更新した後に、[OK] を選択して変更を保存します。次に、変更を有効にするために、実行中の RoboServer を再起動します。

RoboServer ID

それぞれの RoboServer には、**Management Console** > **[管理]** > **[RoboServer]** > **[サーバー]** の下に表示される一意の RoboServer ID があります。

この ID を変更するには、`roboserver.settings` 設定ファイル内の `roboserver-id` 変数を編集します。

ホストおよびポート設定

クラウド内や Docker コンテナ内で NAT を使用して実行している場合など、検出されたホスト アドレスが実際のアドレスと異なる場合は、正しい名前または IP アドレスとポート番号を指定します。

RAM 使用量の調整

RoboServer で使用される RAM の最大量を変更するには、「RAM の割り当ての変更」を参照してください。

コマンド ラインの設定パラメータ

RoboServer では、次のような設定パラメータが受け入れられます。

コンテナ化された環境では、環境変数を使用してパラメータを設定します。「コンテナベースの展開」を参照してください。

パラメータ	説明
`-mcUrl <arg>` (必須)	登録用の Management Console の URL を以下の形式で指定します: `http[s]://<hostname>:<port number>`。 例: `-mcUrl http://localhost:50080/ManagementConsole` Document Transformation ステップと [コールバック] オプションを持つロボットで、RoboServer のホスト名または IP アドレスを使用します。 'localhost' を使用している場合、Document Transformation サービスは Management Console に接続できず、コールバック ロボットは実行されません。

`-ss`
`--mcSharedSecret <MC shared secret>`
(必須)
	Management Console で RoboServer を認証するための共有シークレットを指定します。
	Management Console サービスの認証セクションから共有シークレットをコピーします。詳細については、『Tungsten RPA のヘルプ』の「サービス認証」を参照してください。
	共有シークレットを設定するには、RoboServer 設定アプリケーションを使用します。

`-cl`
`--cluster <arg>`
(必須)
	RoboServer を Management Console 内の指定されたクラスタに自動的に登録します。
	この例では、RoboServer が本番クラスタに RoboServer 自体を登録します。
	• `-cl Production`
	• `-mcUrl http://localhost:50080/ManagementConsole -ss [MC 共有シークレット] -cl Production`

`-eh` `--externalHost <外部 IP アドレス/ホスト名>`	RoboServer ホストの名前または IP アドレスを明示的に指定します。 クラウド内または Docker コンテナ内で NAT を使用して実行している場合など、実際のホスト アドレスがローカルで検出されたものと異なる場合に、このパラメータを使用します。 例: `-eh 10.10.0.123`
`-ep` `--externalPort <ポート番号>`	RoboServer ホストのポート番号を明示的に指定します。 クラウド内または Docker コンテナ内で NAT を使用して実行している場合など、実際のホスト アドレスがローカルで検出されたものと異なる場合に、このパラメータを使用します。

パラメータ	説明
`-resolveHostToIp`	RoboServer と Management Console 間のソケット接続にのみ関連します。 デフォルトでは `false` に設定されています。`true` に設定すると、RoboServer のホストがその IP に解決されます。
`-jmxPass`	JMX アプリケーションで RoboServer を監視するための JMX パスワードを設定します。

`-v`
`--verbose`
(任意)
	出力ステータスとランタイム イベントを表示します。

`-V`
`--version`
(任意)
	RoboServer のバージョン番号を表示して終了します。

`-h` `--help`	ヘルプを表示します。
`-pauseAfterStartupError` (任意)	起動中にエラーが発生した場合に、RoboServer を一時停止します。
`-s` `--service <service-name:service-parameter>`	RoboServer を起動するための RQL または JMX サービスを指定します。 このパラメータは少なくとも一度指定する必要があり、同じ RoboServer で複数のサービスを開始するために複数回使用されることがあります。利用可能なサービスは、インストールによって異なります。 例：`--service socket:50000` 例：`--service jmx:50100` 詳細については、下の表で「利用可能なサービス」を確認してください。
`-p` `--port <ポート番号>`	`-s socket:<port number>` を呼び出すための省略形 例: `--port 50000`
`-sslPort <ポート番号>`	`-s ssl:[ポート番号]` を呼び出すための省略形 例: `--sslPort 50000`

`-nd`
`--NoDoc`
(任意)
	この RoboServer へのロボット ドキュメント要求を禁止します。
	詳細については、『Tungsten RPA のヘルプ』の「ドキュメントを生成」を参照してください。

`-sn`
`--serverName`
(任意)
	Tungsten RPA で Analytics の統計を記録するためのサーバー名を RoboServer に設定します。指定しない場合は、サーバーの IP アドレスに基づいて統計が収集されます。

パラメータ	説明
-ll --licenseLimit <limit> (任意)	RoboServer が受け取ることができるライセンス ユニットの最大数を指定します。
-client	Management Console クラスタのクライアントとして RoboServer を実行します。 高可用性モードではサポートされていません。

利用可能なサービス

サービス	説明
--service socket:<port number>	RoboServer が Management Console クラスタ向けのサービスとして実行されるように指定します。 <portNumber>: 監視対象のソケット サービスのポート番号。
--service ssl:<port number>	RoboServer を Management Console クラスタ向けのサービスとして実行し、セキュアな接続を作成することを指定します。 <portNumber>: 監視対象のソケット サービスのポート番号。
--service jmx:<port number>,<RMI URL>	<jmx_port_Number>: 監視対象の JMX サービスのポート番号。 <jmx_rmi_url>: JMX サービス向けのオプションの RMI ホストとポート。ファイアウォール経由で接続するために使用します。 例: --service jmx:example.com:51001

RoboServer と Management Console 間の接続タイプを設定するには、次のいずれかのパラメータを使用します。

- -client
- --service socket:<port number>

RoboServer を接続するコマンド ラインの例:

- Management Console クラスタ向けのクライアントとして:

  ```
  -client -mcUrl http://localhost:50080/ManagementConsole -cluster <client cluster> -
  ss <MC shared secret>
  ```

  ここで、clientCluster は、Management Console で作成されたクライアント接続タイプのクラスタの名前です。

- Management Console クラスタ向けのサービスとして:

  ```
  -service socket:50000 -mcUrl http://localhost:50080/ManagementConsole -cluster
   <service cluster> -ss <MC shared secret>
  ```

  ここで、<serviceCluster> は、Management Console で作成されたサービス接続タイプのクラスタの名前です。

セキュリティ

RoboServer 設定アプリケーションの **[セキュリティ]** タブで、RoboServer TLS 設定、一般的なセキュリティ制限、および監査ログの設定を指定します。

RoboServer は、実行されているコンピュータでファイルを作成および編集できるようになります。

組み込み Derby データベースを使用を使用している場合、ロボットはコンピューター上でファイルの作成および編集を行うことができます。お使いのネットワーク環境では、MySQL または別のエンタープライズクラス データベースを使用することをお勧めします。

Connector の使用を許可

RoboServer で実行している場合、この設定により、RoboServer が実行されているコンピュータ上のロボットでカスタムの Connector を使用できるようになります。ロボットの [カスタム アクション] ステップでカスタム コネクタを使用します。詳細については、『Tungsten RPA のヘルプ』を参照してください。

Management Console から JDBC ドライバーを受け入れる

JDBC ドライバーを Management Console から RoboServer へ分配します。

ログ出力ステップでのパスワードの難読化

パスワードを含む変数または式を書き込むときに、[ログ出力] ベーシック エンジン ロボット ステップによって生成されるログ エントリ内のパスワードをマスクするには、このオプションを選択します。

コマンドのタイムアウト

RoboServer がリモート デバイスのコマンドからの応答を待つ時間を指定します。このオプションは、ロボットでのターミナルの自動化および Web サイトのブラウジングにのみ適用されます。

コマンドとは、マウス ボタンをクリックする、アプリケーションを開く、「該当するロケーション」ガードを追加するといった、オートメーション デバイスに送信される命令を意味します。コマンドが指定した時間内に完了できない場合、サービスによって通知が送信され、ロボットの実行が停止します。

ガード チョイス ステップの場合、この設定はワークフローでのガードの呼び出しに適用されますが、ガードのトリガー条件を満たすまでの待機はこのタイムアウトとは無関係であるため、無制限に待機し続ける可能性があります。マウス移動ステップと抽出ステップの使用時に、同様の状況が発生します。コマンドはフィールドで指定されたタイムアウト以内にデバイスで呼び出される必要がありますが、ロボットはコマンドの完了を最大 240 秒間待機します。

JMX サーバーの設定

組み込みの JMX サーバーを使用して、JConsole などのツールを介して実行中の RoboServer を監視します。JConsole を有効にするには、RoboServer コマンド ラインで引数を指定します。

ロボット入力の機密情報の秘匿

[**入力を表示**] オプションにより、ロボットの入力パラメーターを管理インターフェイスに表示するかどうかを制御します。パスワードなどのセキュリティ上の機密情報を非表示にするには、このオプションを使用します。

JMX サーバー アクセス

デフォルトでは、サーバー上の正しいポートにアクセスできるすべてのクライアントが JMX サーバーにアクセスできます。認証を要求するには、**[パスワード使用]** オプションを選択します。有効化すると、接続時に、指定したユーザー名とパスワードの入力を求められるようになります。

ハートビート通知

0 秒を超える間隔を指定すると、RoboServer が実行されており、クエリに応答している場合に、JMX サーバーはその間隔でハートビート通知を送信します。

本番構成

安定した高性能な本番環境を確保するには、必要に応じてデフォルトの RoboServer のパラメータを調整します。次のような設定オプションを使用することができます。

- RoboServer インスタンスの数
- 同時に実行されるロボットの数
- メモリの割り当て
- RoboServer のスケーリングの使用

RoboServer インスタンスの数

RoboServer は Java 仮想マシン (JVM) 上で実行され、JVM 自体はオペレーティング システムと基盤となるハードウェア上で実行されます。JVM とオペレーティング システムは定期的に更新され、ハードウェアは進化し、新しいバージョンごとにパフォーマンスの向上が図られます。一般的なパフォーマンス ガイドラインを提供することはできますが、テストを行うことが、最適な設定を決定するための唯一の方法となります。

一般的なルールとして、2 つの RoboServer インスタンスを実行すると、パフォーマンスが向上する可能性があります。JVM はメモリ管理にガベージ コレクション (GC) を使用します。ほとんどのハードウェアでは、GC は単一の CPU コアのみを使用するため、クアッドコア プロセッサでは CPU の 75% がアイドル状態になります。2 つの RoboServer インスタンスを使用すると、1 つのインスタンスが GC を実行している間に、もう 1 つのインスタンスは CPU を使用し続けることができます。ただし、ガベージ コレクタの CPU 使用率は JDK のバージョンに依存し、環境によっては GC が複数の CPU コアを使用する場合もあります。

同時に実行されるロボットの数

RoboServer が同時に実行できるロボットの数は、使用可能な CPU 容量と、処理に必要なデータを取得できる速度に応じて異なります。この値は Management Console のクラスタ設定で設定します。

遅い Web サイトとやり取りするロボットは、応答の速い Web サイトとやり取りするロボットよりもはるかに少ない CPU を消費します。CPU 使用率は次のように表現することができます。

```
CPU（コア）%= 1-WaitTime / TotalTime
```

たとえば、ロボットの実行に 20 秒かかり、Web サイトの応答を待つために 15 秒かかる場合、実際に実行されているのは 5 秒だけです。この 20 秒全体で見ると、これは CPU コアの平均 25% に相当します。ロボットのステップは順番に実行されるため、1 台のロボットが 1 度に使用できる CPU コアは 1 つだけです。クアッドコア CPU では、5 秒間実行して 15 秒間待機するロボットは、合計 CPU 容量の約 6% を効率的に使用します。

デフォルトでは、RoboServer は 20 台のロボットを同時に実行するように設定されています。各ロボットが約 6% の CPU を使用した場合、約 16 〜 17 台のロボットで CPU 使用率が最大になります。このようなロボットを 33 台起動すると、RoboServer に過負荷がかかり、それぞれのロボットの完了に 2 倍の時間がかかることになります。実際には、ロボットの CPU 使用率は、ロボットのロジックとデータのやり取りを行う Web サイトに応じて、広範囲に及ぶ可能性があります (CPU コアの 5% 〜 95%)。その結果、信頼性を保って正しい最大値は予測することはできません。負荷を増加しながら RoboServer の CPU 使用率とロボットの実行時間を監視して負荷テストを行うというのが最大値を判断するための唯一の方法です。

メモリ割り当て

メモリの可用性は、RoboServer がサポートする同時ロボットの数にも影響を及ぼします。メモリ使用量はロボットによって異なり、数メガバイトから数百メガバイトの範囲になります。メモリの割り当てを調整する方法については、「RAM の割り当ての変更」を参照してください。

JVM はオペレーティング システムからのメモリの予約を行いますが、メモリが使用された場合、そのメモリが解放され OS により再び利用可能になることはありません。正しいメモリ割り当てを確保するには、負荷テスト中にメモリ使用量を監視します。CPU 使用率を 100% にするテストを実行し、JVM (java.exe) が実際に使用した予約メモリの量を確認します。すべての予約メモリが消費された場合は、割り当てを 2 倍に増やしてテストを繰り返します。最終的に、JVM は予約されたメモリをすべて使用することはなく、その時点で、使用されるメモリの量に実際の要件が反映されるため、RoboServer 用に設定する必要があります。

RoboServer のスケーリングの使用

デフォルトの方法は負荷分散スケーリングです。この方法では、同時に実行されるロボットに使用可能な空き実行スロットの数が優先されます。2 つの RoboServer に同じ数の空きスロットがある場合は、キューのサイズを使用して、どちらが次のロボットを受け取るかが決定されます。

オンサイト展開の場合、システムで空き実行スロット数が最も多い RoboServer でロボットがキュー待ち状態となり、すべての RoboServer 間で作業が均等に分散されます。

マルチテナント クラウド展開では、別の方法が必要になる場合があります。スケーラブルな方法では、実行スロットが利用可能な任意の RoboServer でロボットがキュー待ち状態となり、残りのスロットが最も少ないロボットが優先されます。これにより、未使用の RoboServer をより迅速にスケール ダウンできるようになります。この方法によって、RoboServer の最大数に達しておらず、シャットダウン モードの RoboServer がない場合に、スケーラビリティが最適化されます。

スケーリング方法は起動時に設定する必要があります。Management Console の実行中に方法を変更するには、再起動が必要です。

スケーラブルな方法を設定するには、Management Console の `application.properties` ファイルで次のプロパティを設定します。

```
mc.robo-servers.robot-distribution-strategy=scalable
```

RAM の割り当ての変更

インストール時に、RoboServer はデフォルトで最大 8 GB という RAM の割り当てで設定されます。これは通常、一般的なワークロードには十分なサイズですが、多数のロボットを並行して実行する場合や大量のメモリを使用するロボットがある場合は、この制限を増やす必要があることがあります。

RoboServer のメモリの割り当てを変更するには、次の手順を実行します。

1. `[インストール ディレクトリ]/bin/` に移動し、テキスト エディターで `RoboServer.conf` ファイルを開きます。
2. `wrapper.java.maxmemory` パラメーターを含む行を見つけます。
   行が存在しない場合は、追加してください。
3. 先頭の # を削除して行のコメントを解除し、その値を編集します。
   たとえば、RoboServer で最大 16GB の RAM を使用するには、以下を入力します。`wrapper.java.maxmemory=16384`

RoboServer のシャットダウン

RoboServer を停止するには、次の手順を実行します。

1. [Management Console] > [管理] > [RoboServer] の順に移動します。
2. RoboServer のコンテキスト メニューから **[RoboServer を停止]** を選択します。
   サーバーを停止する方法のオプションを含むダイアログ ボックスが表示されます。現在実行中のロボットの処理方法を選択することもできます。詳細については、『Tungsten RPA のヘルプ』の「RoboServer」を参照してください。

RoboServer サービスの開始のトラブルシューティング

サービスが開始されない場合は、Windows イベント ログで RoboServer メッセージを探します。

`wrapper.syslog.loglevel=INFO` 引数を使用してサービスがインストールされていることを確認してください。

詳細については、『Tungsten RPA インストール ガイド』を参照してください。

第 4 章

Kapplets

Tungsten RPAKapplets は、ロボットを実行するための Web インターフェイスを提供するアプリケーションです。

Tungsten RPA には、Kapplets の実行に必要な、本番環境対応の Java ランタイム環境と組み込みの Apache Tomcat ライブラリが含まれています。

デフォルトでは、Kapplets サービスは非本番環境のプラットフォーム データベースとして H2 データベースを使用します。本番環境での展開では、Kapplets を本番レベルのデータベースに接続し、必要な接続設定とセキュリティ設定を完了します。

Kapplets を設定および管理するには、次のトピックを参照してください。

- Kapplets の起動
- Kapplets の設定
- データベースの設定
- Management Console での認証
- データの暗号化
- RAM の割り当て
- ログ記録の設定

Kapplets の開始

次のいずれかの方法で Kapplets を起動します。

- Windows のスタート メニューから Kapplets プログラムを選択する。
- コマンド ラインから Kapplets 実行ファイルを実行する。

Kapplets を Windows サービスとして実行するように設定するか、コンテナ化された環境で実行するように設定することもできます。

Kapplets の構成

次の場所にある `.properties` ファイルを使用して Kapplets を設定します。

`[インストール_ディレクトリ]/WebApps/kapplets`

このフォルダには次の 2 つの設定ファイルが含まれています。

ファイル名	説明
`application.properties`	サーバー ポート、データベース接続、および SSL などの主要なアプリケーションの設定を行います。「設定プロパティ」を参照してください。
`log4j2.properties`	高度なログの設定を行います。「ロギングの設定」を参照してください。

設定プロパティ

Kapplets の設定ファイル `application.properties` には、次のプロパティが含まれています (わかりやすくするためにここではグループ化しています)。必要に応じて、このプロパティの設定を行います。たとえば、ポート 8080 で Kapplets を実行するには、`server.port=8080` と設定します。

コンテナ化された環境では、環境変数を使用してパラメータを設定します。「コンテナベースの展開」を参照してください。

時間ベースの設定を含むプロパティには、持続時間形式の数値を入力することができます。次のような接尾辞付きの簡単な表記を使用します: 秒は `s`、分は `m`、時間は `h`、日は `d`。

実行

プロパティ	説明
`kapplets.execution.result.encryption.enabled`	機密データを保護するためにパスワード フィールドの暗号化を可能にします。 デフォルト値: `true`
`kapplets.execution.result.encryption.key`	パスワード フィールドの暗号化キー。セキュリティを強化するためにデフォルト値から変更します。 デフォルト値: `5d0f68b7-ac91-47fe-8155-25674c12140a`
`kapplets.execution.result.encryption.salt`	パスワード フィールドの暗号化ソルト。セキュリティを強化するためにデフォルト値から変更します。 デフォルト値: `9c461389-fc0d-4439-af40-15b5c395a148`
`kapplets.execution.result.file-date-format`	エクスポートされたファイル名に含まれる日付フォーマット パターン。ファイル名にのみ影響し、ファイルの内容に影響を及ぼすことはありません。 デフォルト値: `yyyy-MM-dd_HH-mm-sss`
`kapplets.execution.result.xls-export-formats*`	エクスポートされた Excel ファイルのフォーマットを設定するためのプロパティ セットで、日付や数字のロケール固有の表示が含まれます。
`kapplets.execution.purge.cron`	古い実行結果を日次や週次などで削除するためのタイミングを定義する、cron 形式の式を指定します。 デフォルト値: `0 0 3 * * *` (毎日午前 3 時)

プロパティ	説明
kapplets.execution.purge.timeout	実行結果レコードが保持される期間を指定します。この期間を超えた記録は、スケジュールされた次回のパージで削除されます。 デフォルト値: 365d
kapplets.execution.watcher.check-interval	Kapplet の実行が、許可された期間を超えた場合に終了する必要があるかどうかを確認する間隔。 デフォルト値: 30m
kapplets.execution.watcher.timeout	Kapplet に「失敗」というマークを付けるまでの最大実行時間。 デフォルト値: 190m

オフライン ドキュメント

プロパティ	説明
kapplets.help.offline-base-url	オフライン ドキュメントを含むフォルダへのパス。 位置情報の URL の先頭の部分は、file:/// である必要があります。例: kapplets.help.offline-base-url= file:///c:/TungstenRPAManagementConsoleDocumentation

KeyStore の設定

プロパティ	説明
kapplets.key-store.current.location	現在の Java KeyStore ファイルへのパス。このファイルにより、機密データを暗号化するために使用した証明書を安全に保存し、次にその証明書をデータベースに保存します。パスをファイル システムの位置として指定します。例: file:/c:/certs/kapplets.p12 デフォルト値: classpath:kapplets.p12 (<INSTALL DIR>/WebApps/kapplets.war/WEB-INF/classes/kapplets.p12)
kapplets.key-store.current.alias	現在の Java KeyStore ファイル内の証明書を参照するために使用されるエイリアス。 デフォルト値: kapplets
kapplets.key-store.current.password	現在の Java KeyStore ファイルにアクセスしてロックを解除するためのパスワード。 デフォルト値: changeit
kapplets.key-store.previous.location	以前の Java KeyStore ファイルへのパス。パスをファイル システムの位置として指定します。例: file:/c:/certs/kapplets.p12
kapplets.key-store.previous.alias	以前の Java KeyStore ファイル内の証明書を参照するために使用されるエイリアス。

プロパティ	説明
kapplets.key-store.previous.password	以前の Java KeyStore ファイルにアクセスしてロックを解除するためのパスワード。

デフォルトの言語

プロパティ	説明
kapplets.language.default	Kapplets のデフォルトの UI 言語です。 デフォルト値: en

リポジトリ同期

プロパティ	説明
kapplets.repo.cache.check-interval	内部 Kapplets リポジトリを Management Console リポジトリと同期する必要があるかどうかを確認する間隔。 間隔が短すぎると、Management Console とデータベースが過負荷がかかることがあります。 デフォルト値: 1m

サービス接続

プロパティ	説明
kapplets.service-connection.internal-mc-url	パブリック/プライベート ネットワーク (Docker など) がある環境で Management Console に接続するための追加の内部 URL。設定されていない場合、この値はデフォルトの kapplets.service-connection.mc-url になります。
kapplets.service-connection.mc-url	Management Console に接続するための URL。 デフォルト値: http://localhost:50080
kapplets.service-connection.kapplets-url	Management Console での認証中の HTTP コールバックの URL。 デフォルト値: http://localhost:50081
kapplets.service-connection.shared-secret	Management Console でのサービス認証用の共有シークレット。
kapplets.service-connection.shared-secret-file	Management Console のサービス認証に使用される共有シークレットを含むファイルへのパス。両方が指定されている場合は、kapplets.service-connection.shared-secret が優先されます。

Spring Boot のプロパティ

Kapplets サービスは Spring Boot 上に構築されているため、Kapplets アプリケーションのプロパティとともに、次の組み込み Spring Boot プロパティを変更できます。

プロパティ	説明
`server.port`	サーバーの HTTP ポート。 デフォルト値: `50081`
`server.ssl.*`	HTTPS 経由で Kapplets をホストするためのプロパティのグループ。 詳細については、「HTTPS 上の Web ベースのコンポーネントのホスティング」を参照してください。
`spring.datasource.*`	データベースへの接続を設定するためのプロパティのグループ。

Management Console での認証

プロジェクトとロボット データを取得し、ロボットを実行して、Kapplets 内でユーザーを認証するには、Kapplets を Management Console で認証する必要があります。

Kapplets を起動する前に、次の必須のプロパティが設定されていることを確認してください。

- `kapplets.service-connection.mc-url` - Management Console の URL。
- `kapplets.service-connection.kapplets-url` - Kapplets サービスの URL。

さらに、共有シークレットを設定し、それらのシークレットが Management Console 側および Kapplets 側で一致している必要があります。

Management Console では、共有シークレットは `mc.auth.shared-secrets.kapplets` プロパティを使用して設定されます。事前に設定されていない場合は、Management Console で自動的に生成されます。シークレットには、[管理] > Kapplets 用の [サービス認証を使用] の下の Management Console からアクセスします。

Kapplets で、共有シークレットのプロパティを次のように設定します。

```
kapplets.service-connection.shared-secret
```

共有シークレットが設定されていない場合は、Kapplets ユーザー インターフェイスにシークレットを入力するように求めるメッセージが表示されます。

正しい共有シークレットがない場合、Kapplets は Management Console に接続できず、機能しません。

データベースの設定

Kapplets は Management Console と同じデータベース アクセスおよび初期化モデルを使用しますが、以下にいくつかの相違点を示します。デフォルトでは、非本番環境のプラットフォーム データベースと

して H2 データベースが使用されます。このデータベースはテストおよび開発のみを目的としています。すべての H2 データベース ファイルは、以下の場所にローカルで保存されています。

- Windows: `C:\Users\[ユーザー]\AppData\Local\Tungsten RPA\[バージョン ]\Data`
- Linux: `/home/[ユーザー]/.Tungsten RPA/[バージョン ]/Data`

デフォルトのフォルダを変更するには、データベース接続の設定を変更してください。

以下のような相違点があります。

ログ記録、分析、または収集データベースはありません

Kapplets ログ記録、分析、または収集データベースは使用されません。これらのコンポーネントに関する設定は無視してください。

SQL Server 固有のパラメータ

Kapplets を SQL Server に接続するときに、`spring.datasource.url` プロパティにカーソル選択パラメータを追加します。

```
spring.datasource.url=jdbc:sqlserver://database-
service:1433;databaseName=database-name;SelectMethod=cursor
```

初期化スクリプト

Kapplets の場合は、次の場所にあるスクリプト サブディレクトリを使用します。

`[インストール_ディレクトリ]/documentation/sql`

- `kapplets`: データベース プラットフォームごとに整理された、Kapplets 用の SQL スクリプト。
- `quartz`: データベース プラット フォームごとに整理された、Quartz のスケジューリング スクリプト。

データの暗号化

Kapplets のデータ暗号化メカニズムは、「Management Console のデータ暗号化」セクションで説明されている Management Console と同じように動作します。

デフォルトでは、Kapplets のインストールに含まれている KeyStore を使用します。RSA 非対称キー ペアを組み合わせた独自の KeyStore の提供を強くお勧めします。

KeyStore を設定するには、Kapplets の `application.properties` ファイルで以下のプロパティを指定します。

- `kapplets.key-store.current.*`- 現在の KeyStore 設定を定義します。
- `kapplets.key-store.previous.*`- KeyStore のアップグレード時にのみ使用する以前の KeyStore 設定を定義します。

RAM の割り当て

インストール時に、Kapplets アプリケーションはデフォルトで最大 8 GB という RAM の割り当てで設定されます。これは通常、一般的なワークロードには十分なサイズですが、多数のロボットを並行して実行する場合や大量のメモリを使用するロボットがある場合は、この制限を増やす必要があることがあります。

Kapplets のメモリの割り当てを変更するには、次の手順を実行します。

1. `[インストール ディレクトリ]/bin/` に移動し、テキスト エディターで `Kapplets.conf` ファイルを開きます。
2. `wrapper.java.maxmemory` パラメーターを含む行を見つけます。
   行が存在しない場合は、追加してください。
3. 先頭の # を削除して行のコメントを解除し、その値を編集します。
   たとえば、Kapplets が最大 16 GB の RAM を使用できるようにするには、次のように入力します: `wrapper.java.maxmemory=16384`。

ロギングの設定

ログの記録を調整して、問題のトラブルシューティングやシステムのパフォーマンスを監視します。

Kapplets はログ記録に Log4j2 フレームワークを使用します。アプリケーション ログ ファイルはアプリケーション データ ディレクトリ内の [ログ] フォルダに保存されます。これらのフォルダの詳細については、『Tungsten RPA インストール ガイド』を参照してください。

基本設定

ログ記録に対する変更は、多くの場合、次の場所にある `application.properties` ファイルへのプロパティの追加によるログ記録レベルの調整のみとなります

`[インストール_ディレクトリ]/WebApps/kapplets`

たとえば、Quartz のデバッグ レベルのログ記録を有効にするには、次のプロパティを追加します: `logging.level.org.quartz=debug`。

詳細については、Spring Boot のログ設定に関するドキュメントを参照してください。

高度な設定

高度なログ設定を調整するには、次のファイルを編集してください:

`[インストール_ディレクトリ]/WebApps/kapplets/log4j2.properties`

詳細については、Log4j2 のドキュメントを参照してください。

第 5 章

ロボット ファイル システム

Tungsten RPA ロボット ファイル システム (RFS) は、ロボットがプロセス中にデータの読み取りと書き込みに使用する、設定可能な共有ファイル ストレージです。

RFS サーバーは、RoboServer、Design Studio インスタンス、および Desktop Automation エージェントに共有ストレージを提供します。

ロボット ファイル システムは SMBv1 プロトコルをサポートしていません。

ロボット ファイル システムにアップロードまたはダウンロードできるファイルの最大サイズは、使用可能なメモリによって制限されます。

Tungsten RPA には、本番環境対応の Java ランタイム環境と組み込みの Apache Tomcat ライブラリが含まれているため、ロボット ファイル システムを実行するためにそれらを個別にインストールする必要はありません。

RFS を本番環境用に準備するには、ファイル ストレージを設定し、必要な接続とセキュリティ オプションのセットアップを行います。

- ロボット ファイル システムの起動
- ロボット ファイル システムの設定
- ロボット ファイル システム サーバーのセットアップ
- RAM の割り当て
- ログ記録の設定

ロボット ファイル システムの起動

ロボット ファイル システムを起動するには、次の手順を実行します。

- Windows のスタート メニューからロボット ファイル システム プログラムを選択する。
- コマンド ラインからロボット ファイル システムの実行ファイルを実行する。

また、Windows サービスとして、またはコンテナ化された環境で実行するように設定することができます。

ロボット ファイル システムの設定

ロボット ファイル システムは次の場所にある設定ファイルを使用します。

`[インストール_ディレクトリ]/WebApps/rfs`

このフォルダには次の 2 つの設定ファイルが含まれています。

ファイル名	説明
`application.properties`	サーバー ポート、データベース接続、および SSL などの主要なアプリケーション設定。「設定プロパティ」を参照してください。
`log4j2.properties`	高度なログ設定。「ロギングの設定」を参照してください。

設定プロパティ

ロボット ファイル システムの設定ファイル `application.properties` には、次のプロパティが含まれています (わかりやすくするためにここではグループ化しています)。必要に応じて、このプロパティの設定を行います。たとえば、ポート 8080 でロボット ファイル システムを実行するには、`server.port=8080` と設定します。

コンテナ化された環境では、環境変数を使用してパラメータを設定します。「コンテナベースの展開」を参照してください。

絶対パス

プロパティ	説明
`rfs.allow-absolute-paths`	`true` に設定した場合は、絶対パス (`C:\files` など) を使用した RFS ファイル共有が許可されます。 `false` に設定し、`rfs.data-path` を `/data` に設定した場合は、`/data` フォルダ内の共有のみが許可されます。 デフォルト値: `false`

フォルダの作成

プロパティ	説明
`rfs.create-folders`	RFS がフォルダを自動的に作成できるようにします。 デフォルト値: `true`

一時データのフォルダ

プロパティ	説明
`rfs.data-path`	一時的なロボット実行データを保存するフォルダを指定します。例: `C:/RFSData`。 `rfs.allow-absolute-paths=true` の場合にのみ絶対パスを指定できます。一時的な共有は、指定されたフォルダのサブフォルダとして作成および削除されます。

サービス接続

プロパティ	説明
`rfs.service-connection.mc-url`	Management Console の接続 URL。 デフォルト値: `http://localhost:50080`
`rfs.service-connection.shared-secret`	Management Console でのサービス認証に使用される共有シークレット。
`rfs.service-connection.shared-secret-file`	Management Console のサービス認証用の共有シークレットを含むファイルへのパス。 両方が設定されている場合、このプロパティは `rfs.service-connection.shared-secret` よりも優先されます。

Spring Boot のプロパティ

ロボット ファイル システムは Spring Boot をベースにして構築されています。これは、RFS アプリケーション プロパティとともに、次のような組み込みの Spring Boot プロパティも変更できることを意味します。

プロパティ	説明
`server.port`	サーバーの HTTP ポート。 デフォルト値: `50082`
`server.ssl.*`	HTTPS 経由で RFS をホストするためのプロパティのグループ。 詳細については、「HTTPS 上の Web ベースのコンポーネントのホスティング」を参照してください。

ロボット ファイル システム サーバーのセットアップ

ロボット データに対するファイル共有と保存を有効にするために RFS サーバーを設置します。

1. テキスト エディターで、次の場所にある `application.properties` ファイルを開きます: `[インストール_ディレクトリ]/WebApps/rfs`。

2. `rfs.service-connection.mc-url` プロパティに Management Console の URL を指定します。
   例: `rfs.service-connection.mc-url=http://localhost:50080`
3. 共有シークレットを取得して設定します。
   a. Management Console を開き、スーパーユーザーとしてログインします。
   b. **[管理]** > **[サービス認証を使用]** に移動します。
   c. **[ロボット ファイル システム]** のコンテキスト メニューを開き、**[共有シークレットを表示]** を選択して、共有シークレットをコピーします。次に、共有シークレットを `rfs.service-connection.shared-secret` プロパティに直接貼り付けます。
4. ファイル パスのアクセス権を定義します。
   `rfs.allow-absolute-paths` プロパティを `true` または `false` に設定します。
   - `true` に設定すると、RFS サービス ユーザーがこれらの場所にアクセスできる場合、RFS は Windows の `C:\files` や Linux の `/mnt/data` などの絶対パスを使用できるようになります。
   - `false` に設定すると、アクセスは `rfs.data-path` プロパティで指定されたディレクトリのサブフォルダに制限されます。たとえば、`rfs.data-path=/data` である場合は、`/data` 内のパスのみが許可されます。
5. `rfs.data-path` プロパティで、一時的なロボット実行データを保存するフォルダを指定します。
   例: `rfs.data-path=C:/RFSData`
   `rfs.allow-absolute-paths` が `true` に設定されている場合にのみ絶対パスを使用することができます。一時的な共有は、このディレクトリのサブフォルダとして作成および削除されます。
6. ロボット ファイル システムを起動します。
7. Management Console で RFS サーバーに接続します。
   a. Management Console を開き、**[設定]** > **[一般]** > **[ロボット ファイル システム サーバー]** の順に移動します。
   b. **[ロボット ファイル システム サーバーを使用]** を選択し、`http://localhost:50082` などの RFS サーバー URL を入力します。

ロボットによって使用または生成されるデータを共有および保存するように設定されたファイル システムを使用できるようになりました。ファイル システムの設定を追加するには、『Tungsten RPA のヘルプ』の「ロボット ファイル システム」を参照してください。

例: ロボット ファイル システムへのフォルダのマッピング

この例では、Windows のフォルダを Management Console のロボット ファイル システムにマッピングし、Design Studio のロボットにステップを追加して、このロボット ファイル システムにデータを書き込む方法についての手順を示します。

この例を実行する前に、上記の「ロボット ファイル システム サーバーのセットアップ」の手順と、『Tungsten RPA のヘルプ』の「ロボット ファイル システム」を確認することをお勧めします。これらの手順には、ロボット ファイル システム機能の設定と使用方法に関する詳細情報が記載されています。

この手順は、「ロボット ファイル システム サーバーのセットアップ」の手順が完了していることを前提として記述されています。

1. `application.properties` ファイルで、一時的なロボット実行データを保存するフォルダへのパスを指定します。例:`rfs.data-path=c:/rfs`。

   設定を適用するには、ロボット ファイル システム サーバーを再起動します。
2. **[Management Console]** > **[設定]** > **[一般]** で、**[ロボット ファイル システム サーバー]** を選択します。
3. **[Management Console]** > **[リポジトリ]** > **[ロボット ファイル システム]** で、ファイル システムの設定を追加します。

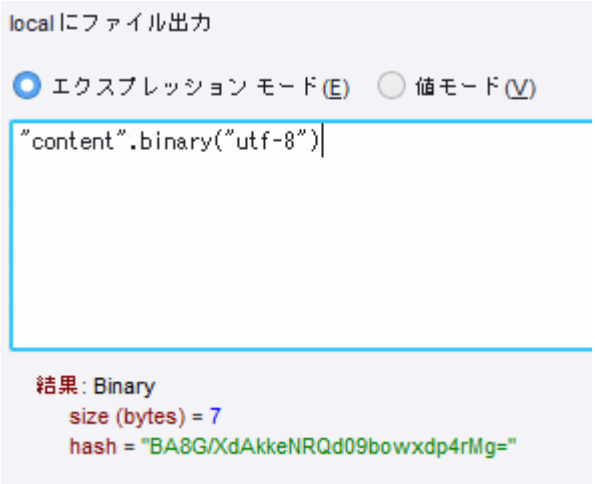
   a. **[一般情報]** タブで、必要なパラメータをすべて指定します。

      [ファイル システム名] パラメータで **RFS1** を指定します。**[パス]** パラメータで、次のように指定します: **rfs1_folder**。

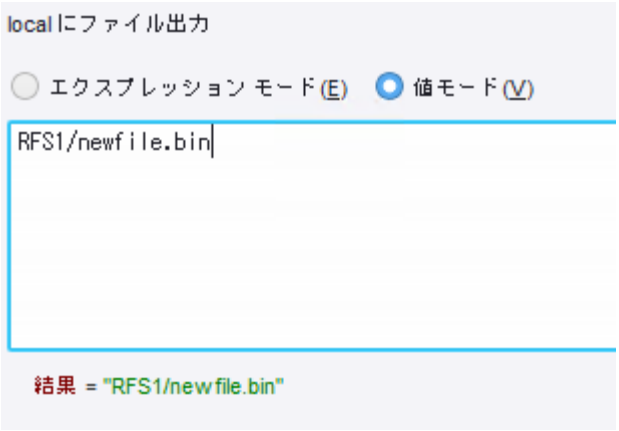
      この場合、`rfs1_folder` は `RFS1` のルート フォルダであるため、絶対パスは `c:/rfs/rfs1_folder` となります。手動で作成しない場合は、`rfs1_folder` が自動的に作成されます。

   b. **[認証されたアクセス トークン]** タブで、ロボットのアクセス トークンを貼り付け、現在のバージョンのロボットのみがファイル システムにアクセスできるようにします (ロボットを Management Console にアップロードする必要があります)。トークンをコピーするには、**Management Console** > **[リポジトリ]** > **[ロボット]** に移動し、必要なロボットのコンテキスト メニューを開いて、**[リソース アクセス トークンを取得]** を選択します。
4. 設定を保存します。
5. Design Studio で、既存のロボットを開くか、新しく作成します。次のプロパティを持つ **[ファイル出力]** ステップを追加します。

コンテンツ: ロボット ファイル システム に書き込むデータを含むバイナリ変数を指定します。この例では、「content」はファイルに書き込まれた文字列です。ファイル コンテンツはバイナリである必要があります。



ファイル名: データを書き込むファイルへのパスを指定します。ロボット ファイル システムの名前は、大文字と小文字が区別されます。



6. ロボットを保存し、Management Console の共有プロジェクトにそのロボットをアップロードします。
7. **[実行の準備]** を選択して、ステップを実行します。
「コンテンツ」のデータが `newFile.bin` ファイルに格納され、`C:/rfs/rfs1_folder/` に保存されます。

一時的な RFS セッション ストレージを設定する

実行プロセス中に、ロボットはロボット ファイル システムでファイルの保存や読み取りを行うことができます。この目的のために、ロボットはロボットのみが使用する一時的な RFS セッション ストレージを作成します。

たとえば、一時的な RFS セッション ストレージは、ロボットが同じような名前で異なる内容のファイルを保存する場合に役立つことがあります。このオプションは、[ファイルの読み取り] や [ファイル出力] など、いくつかの手順で使用できます。

フォルダを一時的な RFS セッション ストレージにマッピングするには、RFS サーバーをセットアップした後に、必要なロボットのステップを設定します。[ファイル名] プロパティで、`robot/newFile.bin` などのパスを指定します。

次に、ファイルは `C:/rfs/8c8a89ca-fd06-4580-99a7-b7ffb7288080` などのフォルダに保存されます。

このフォルダは各ロボットの実行に固有であり、実行の完了後に削除されます。

RAM の割り当て

デフォルトでは、ロボット ファイル システムは最大 8 GB という RAM の割り当てで設定されます。これは通常、一般的なワークロードには十分なサイズですが、多数のロボットを並行して実行する場合や大量のメモリを使用するロボットがある場合は、この制限を増やす必要があることがあります。

ロボット ファイル システムのメモリ割り当てを変更するには、次の手順を実行します。

1. `[インストール_ディレクトリ]/bin/` に移動し、テキスト エディターで `RobotFileSystem.conf` ファイルを開きます。
2. `wrapper.java.maxmemory` パラメーターを含む行を見つけます。
   行が存在しない場合は、追加してください。
3. その行がコメント アウトされている場合は、先頭の # 文字を削除します。その後、値を編集してください。
   たとえば、ロボット ファイル システムが最大 16 GB の RAM を使用できるようにするには、次のように入力します: `wrapper.java.maxmemory=16384`。

ロギングの設定

ログ記録の設定を行って、問題のトラブルシューティングやシステムのパフォーマンスを監視します。

ロボット ファイル システムは、ログ記録に Log4j2 フレームワークを使用します。アプリケーション ログ ファイルはアプリケーション データ ディレクトリ内の [ログ] フォルダに保存されます。これらのフォルダの詳細については、『Tungsten RPA インストール ガイド』を参照してください。

基本設定

多くの場合、ログ記録に対する変更を行うには、ログ レベルの調整のみを行います。次の場所にある `application.properties` ファイルにプロパティを追加します。

`[インストール_ディレクトリ]/WebApps/rfs`

たとえば、Quartz に対してデバッグ レベルのログ記録を有効にするには、次のプロパティを追加します: `logging.level.org.quartz=debug`

詳細については、Spring Boot のログ設定に関するドキュメントを参照してください。

高度な設定

高度なログ設定を調整するには、次のファイルを編集してください:

`[インストール_ディレクトリ]/WebApps/rfs/log4j2.properties`

詳細については、Log4j2 のドキュメントを参照してください。

第 6 章

TLS による安全な通信

Tungsten RPA には、HTTPS 経由で Web ベースのコンポーネントをホストし、RPA コンポーネント間で安全なクライアント サーバー TLS 通信を確立するための方法が用意されています。この通信では、証明書を使用して、当事者間で送信されるデータを暗号化します。

次のトピックでは、TLS を使用した安全な通信を設定および管理する方法について説明します。

- 証明書の信頼性と設定
- サーバー側の設定
- クライアント側の設定

証明書の信頼性と設定

RPA コンポーネント間の通信を保護するための証明書は、次の場所から取得できます。

- ブラウザやオペレーティング システムによって信頼され、厳格な ID 検証プロトコルに準拠するパブリック証明機関 (CA)。
- 内部展開で使用されることが多いプライベート CA。

証明書を生成および管理するプロセスは、組織のポリシーとインフラストラクチャに応じて異なり、さまざまなツールと方法を使用して完了することができます。

自己署名証明書はテスト環境や非商用環境では使用できますが、本番環境での使用は推奨されません。自己署名証明書では十分なセキュリティ レベルが提供されません。

自己署名証明書を使用する場合は、CA ルート証明書の代わりに、クライアントのトラストストアにサーバー証明書をインポートします。

サーバー側の設定

RPA コンポーネントを HTTPS/TLS 経由でサーバーとして動作するように設定するプロセスは、コンポーネントに応じて異なります。設定手順は次のようにグループ化されます。

- HTTPS 経由での Web ベースのコンポーネントのホスティング
- TLS 経由での RoboServer のホスティング
- TLS 経由での Desktop Automation サービスのホスティング

HTTPS 経由での Web ベースのコンポーネントのホスティング

PEM でエンコードされた証明書、または JKS あるいは PKCS12 形式で保存された信頼情報を使用して、HTTPS 経由で Management Console、Kapplets、およびロボット ファイル システムをホストします。

PEM エンコードされた証明書の使用

各コンポーネントの `application.properties` ファイルで、適切なプロパティを使用して、証明書と秘密鍵への完全なファイル パスを指定します。環境とオペレーティング システムに合わせて、必要に応じてパスを調整してください。

例:

```
server.ssl.bundle=webserver
spring.ssl.bundle.pem.webserver.keystore.certificate=file:/c:/ssl/app.crt
spring.ssl.bundle.pem.webserver.keystore.private-key=file:/c:/ssl/app.key
```

JKS または PKCS12 形式の信頼情報の使用

各コンポーネントの `application.properties` ファイルで、適切なプロパティを使用して、キーのエイリアス、キーストアへの完全なファイル パス、およびそのパスワードを指定します。環境とオペレーティング システムに合わせて、必要に応じて値を調整してください。

例:

```
server.ssl.bundle=webserver
spring.ssl.bundle.jks.webserver.keystore.alias=app
spring.ssl.bundle.jks.webserver.keystore.location=file:/c:/ssl/app.p12
spring.ssl.bundle.jks.webserver.keystore.password=secret
spring.ssl.bundle.jks.webserver.keystore.type=PKCS12
```

TLS 経由の RoboServer のホスティング

安全な環境を実現するには、Management Console クラスタへの接続時に、RoboServer をクライアント接続モードで実行するように設定します。

デフォルトでは、RoboServer が HTTPS 用に設定されている場合、Management Console は安全な通信のために TLS を使用します。この設定では、RoboServer はサーバーとして機能せず、安全な通信が自動的に確立されるため、追加の TLS 設定は必要ありません。

TLS 経由での Desktop Automation サービスのホスティング

Desktop Automation サービスを TLS 経由でホストするには、ユーザー インターフェイスの [証明書] タブで証明書への適切なファイル パスを指定します。

クライアント側の設定

RPA コンポーネントがクライアント サーバー モデルで通信を行う場合、たとえば、クライアント コンポーネントが HTTPS/TLS 経由でサーバー コンポーネントに接続すると、サーバーは TLS ハンドシェイクの一部として証明書を提示します。

安全な接続を確立するために、クライアントはサーバーの証明書を信頼する必要があります。信頼メカニズムは、クライアント アプリケーションとそのアプリケーションが接続するサーバー コンポーネントに応じて異なります。

信頼メカニズムのマトリックス表

クライアント コンポーネント	サーバー コンポーネント			
	Management Console	Desktop Automation サービス	ロボット ファイル システム	SignDoc
Management Console	-	-	JRE トラストストア	-
RoboServer	JRE トラストストア	Node.js トラストストア	Node.js トラストストア	Node.js トラストストア
Design Studio	JRE トラストストア	Node.js トラストストア	Node.js トラストストア	Node.js トラストストア
Kapplets	JRE トラストストア	-	-	-
Desktop Automation サービス	OS トラストストア + カスタム証明書設定	-	OS トラストストア + カスタム証明書設定	-
ロボット ファイル システム	JRE トラストストア	-	-	-
Synchronizer	JRE トラストストア	-	-	-

メカニズムの詳細については、以下のトピックに説明が記載されています。

- JRE トラストストア
- Node.js トラストストア
- OS トラストストア
- Desktop Automation サービス用のカスタム証明書

JRE トラストストア

クライアントが Java Runtime Environment のトラストストアを使用している場合は、TLS 接続の信頼性を確立するために以下のいずれかのオプションを使用します。

- ルート証明書をデフォルトのトラストストアにインポートします。

  発行元 CA のルート証明書を、クライアント アプリケーションが使用している Java KeyStore にインポートします。デフォルトでは、このトラストストアは以下の場所にあります。

  `[インストール_ディレクトリ]/jre/lib/security/cacerts`

  インポートを実行するには、`[インストール_ディレクトリ]/jre/bin` にある `keytool` ユーティリティを使用します。

  例:

  ```
  keytool -import -alias myCA -keystore ..\lib\security\cacerts -trustcacerts
  -file c:\ssl\rootCA.crt
  ```

- カスタムのトラストストアを使用します。

  次の Java オプションを使用してクライアント アプリケーションを設定し、カスタム トラストストアを指定します。

  `javax.net.ssl.trustStore`

  `javax.net.ssl.trustStorePassword`

  これらのオプションは、信頼された証明書を含むトラストストア ファイルを指している必要があります。

  これらのオプションは、`[インストール_ディレクトリ]/bin/<application>.conf` にあるアプリケーション設定ファイル内の `wrapper.java.additional` プロパティで設定します。

  例:

  ```
  wrapper.java.additional.60=-Djavax.net.ssl.trustStore="c:/ssl/rootCA.jks"
  wrapper.java.additional.61=-Djavax.net.ssl.trustStorePassword=secret
  ```

  カスタム トラストストアを使用すると、デフォルトのトラストストアが置き換えられます。デフォルトの信頼された証明書を保持するには、元の `cacerts` ファイルをコピーし、そのファイルにカスタム証明書を追加します。

Node.js トラストストア

Node.js は独自にバンドルされた証明機関のリストを使用します。信頼性を確立するためには、次の手順を実行します。

1. クライアント コンポーネントを実行するために使用されるアカウントを識別します。
2. 発行元 CA のルート証明書をシステムにコピーします。
3. 識別されたアカウントの環境変数 `NODE_EXTRA_CA_CERTS` を追加するようにシステムを設定します。
   - 変数が定義されていない場合は、変数を作成し、その値を証明書ファイルの完全パスに設定します。

- 変数がすでに存在する場合は、参照先のファイルを見つけて、そのファイルにルート証明書の内容を追加します。

4. 識別されたアカウントに、`NODE_EXTRA_CA_CERTS` によって参照されるファイルへの読み取りアクセス権が割り当てられていることを確認します。

OS トラストストア

オペレーティング システムのトラストストアに依存するクライアントは、信頼されたルート証明書を使用して安全な接続を確立します。

自己署名証明書を使用する場合は、その証明書を OS トラストストアにインポートします。

オペレーティング システムによってすでに信頼されているパブリック CA から証明書が発行されている場合、追加の設定は必要ありません。

Desktop Automation サービス用のカスタム証明書

Desktop Automation サービスを HTTPS 経由でホストされている Management Console に接続するには、Desktop Automation サービスのユーザー インターフェイスに移動し、[CA ファイル] フィールドに発行元 CA のルート証明書を指定します。

第 7 章

展開

この章では、Tungsten RPA コンポーネントで使用可能な展開オプションについて説明します。環境の要件に応じて、コンテナへのコンポーネントの展開、Windows サービスとしての実行、またはサイレントインストール手順を使用したインストールおよび削除を行うことができます。

- コンテナベースの展開
- サービスとしての RPA コンポーネントの実行
- Windows でのサイレント インストールとアンインストール

コンテナベースの展開

Tungsten Automation には、コンテナ化された環境に Tungsten RPA を迅速に展開するために役立つさまざまなリソースが用意されています。これには以下のようなものが含まれます。

- RPA コンポーネント用に事前構築された Docker イメージ。
- カスタム Docker イメージの構築に役立つサンプルの Dockerfile。
- 複数コンテナの展開をオーケストレーションするための、サンプルの Docker Compose 設定ファイル。
- Kubernetes 環境に Tungsten RPA を展開するためのサンプルの Helm チャート。

これらのリソースは、単一のマシンで実行する場合、またはクラウドでコンテナをオーケストレーションする場合でも、さまざまな展開シナリオをサポートします。

展開を設定するには、次のトピックを使用してください。

- Docker イメージ
- 環境変数
- Docker Compose の例
- Kubernetesサンプル Helm チャート

Docker イメージ

以下の Linux ベースの事前に構築されたイメージは Docker Hub で公式に公開されています。

コンポーネント	Docker Hub のリンク
Management Console	https://hub.docker.com/r/tungstenautomation/rpa-managementconsole
RoboServer	https://hub.docker.com/r/tungstenautomation/rpa-roboserver

コンポーネント	Docker Hub のリンク
Kapplets	https://hub.docker.com/r/tungstenautomation/rpa-kapplets
ロボット ファイル システム	https://hub.docker.com/r/tungstenautomation/rpa-robotfilesystem
Synchronizer	https://hub.docker.com/r/tungstenautomation/rpa-synchronizer

可能な場合は、事前に構築された Docker イメージを使用してください。これらのイメージは標準的な展開向けに最適化およびテストされており、サポートされたソリューションを提供します。

追加のカスタマイズが必要な場合は、ユーザーがイメージを作成することもできます。`[インストール_ディレクトリ]/docker` フォルダには、事前に構築されたイメージの作成に使用される Dockerfile とサポート リソースが含まれています。必要に応じてこれらのファイルを変更し、環境に合わせてカスタマイズされたイメージを構築します。

環境変数

コンテナ化された環境で、環境変数を使用してコンポーネントを設定します。この方法により、設定ファイルを変更せずにコンテナの動作を動的に調整することができ、CI/CD パイプラインやさまざまなランタイムまたはオーケストレーション プラットフォームを介した柔軟な展開がサポートされます。

- Management Console 変数
- RoboServer 変数
- Kapplets 変数
- ロボット ファイル システムの変数
- Synchronizer 変数

Management Console 変数

対応する環境変数を定義して、Management Console 設定プロパティのリストに記載されている任意のプロパティを設定します。これには、SAML、LDAP、およびカスタム ロールの設定が含まれます。

環境変数で定義されていないプロパティは、デフォルトの設定からその値が取得されます。デフォルト値を上書きするプロパティに対してのみ環境変数を定義する必要があります。

プロパティ名を環境変数に変換するには、次の手順を実行します。

- ドット「.」をアンダースコア「_」に置き換えます。
- ダッシュ「-」を削除します
- プロパティ名のすべての文字を大文字に変換します。
- ロールと権限をカスタマイズする場合は、アクション名を変換しないようにしてください。

例:

```
mc.roles.developer.permissions.vault=viewOAuthClients,viewRobotAccess,viewSecrets
>

MC_ROLES_DEVELOPER_PERMISSIONS_VAULT=viewOAuthClients,viewRobotAccess,viewSecrets
```

標準の設定プロパティとともに、コンテナ化された展開内の Management Console 設定では次の環境変数を使用することができます。

環境変数	説明
JDBC_DRIVER_URL	JDBC ドライバーをダウンロードするための URL を指定します。
JAVA_OPTS	Java 仮想マシン (JVM) のオプションとパラメータを指定します。
SLEEP_DELAY	コンテナが起動する前の遅延を秒数で指定します。

高可用性の展開の場合は、カスタマイズした `hazelcast.yml` ファイルをコンテナにマウントし、このファイルを指すように `SPRING_HAZELCAST_CONFIG` 環境変数を設定します。

RoboServer 変数

環境変数	説明
WRAPPER_MAX_MEMORY	Java ヒープ サイズ (メガバイト (MB))。
WRAPPER_JAVA_ADDITIONAL_<N>	追加の Java パラメータ (<N> は 1 から 6 までの数値です)。
ROBOSERVER_SEC_ACCEPT_JDBC_DRIVERS	リクエストの一部として送信された JDBC ドライバーへのアクセスを許可します。 デフォルト値: false
ROBOSERVER_SEC_LOG_HTTP_TRAFFIC	すべての HTTP トラフィックのログ記録を有効にします。 デフォルト値: false
ROBOSERVER_CONNECTION_TYPE	RoboServer と Management Console 間の接続タイプ。以下のいずれかを指定します。 • ClientConnectionType • SocketConnectionType
ROBOSERVER_SOCKET_SERVICE_PORT	ROBOSEREVER_CONNECTION_TYPE が SocketConnectionType に設定されている場合の RoboServer のポート番号。 デフォルト値: 50000
ROBOSERVER_ENABLE_MC_REGISTRATION	このサーバーを Management Console に登録します。 デフォルト値: false
ROBOSERVER_MC_URL	登録する Management Console の URL。
ROBOSERVER_MC_SHARED_SECRET	Management Console への登録時に使用する共有シークレット。
ROBOSERVER_MC_SHARED_SECRET_FILE	Management Console の登録用の共有シークレットを含むファイルへのパス。

環境変数	説明
ROBOSERVER_MC_CLUSTER	Management Console への登録時に参加するクラスタ名。
ROBOSERVER_EXTERNAL_HOST	ROBOSERVER_CONNECTION_TYPE が SocketConnectionType に設定されている場合に、Management Console がこのサーバーにアクセスするために使用できる外部ホスト名または IP アドレス。 ホストのアドレスがローカル検出と異なる場合にこのパラメータを指定します (Docker またはファイアウォールの展開で一般的)。
ROBOSERVER_EXTERNAL_PORT	ROBOSERVER_CONNECTION_TYPE が SocketConnectionType に設定されている場合に、Management Console がこのサーバーにアクセスするために使用できる外部ポート。 ホストのポートがローカル検出と異なる場合にこのパラメータを指定します (Docker またはファイアウォールの展開で一般的)。
ROBOSERVER_RESOLVE_TO_IP	ROBOSERVER_CONNECTION_TYPE が SocketConnectionType に設定され、ROBOSERVER_EXTERNAL_HOST が設定されていない場合に、RoboServer による IP アドレスへのホスト名の解決を許可します。Kubernetesで RPA を展開する場合はこのパラメータを有効にしてください。 デフォルト値: false
ROBOSERVER_SERVER_NAME	ログ内の RoboServer を識別するため、および RPA の Analytics を識別するために使用する名前。指定しない場合は IP アドレスが使用されます。
ROBOSERVER_JMX_ENABLE	JMX サービスを有効にします。 デフォルト値: true
ROBOSERVER_JMX_PORT_NUMBER	JMX サービスのポート番号。 デフォルト値: 50100
ROBOSERVER_JMX_SHOW_INPUTS	JMX を通じてロボットの入力を公開します。 デフォルト値: true
ROBOSERVER_JMX_HEARTBEAT_INTERVAL	ハートビートの通知間隔 (秒)。 デフォルト値: 0
ROBOSERVER_JMX_ENABLE_AUTHENTICATION	JMX リクエストの認証を有効にします。 デフォルト値: false
ROBOSERVER_JMX_USERNAME	JMX 認証用のユーザー名。
ROBOSERVER_JMX_PASSWORD	JMX 認証用のパスワード。
ROBOSERVER_JMX_RMI_ENABLE	JMX 用の RMI を有効にします。 デフォルト値: false

環境変数	説明
ROBOSERVER_JMX_RMI_URL	JMX サービスの RMI ホストとポート。例: 0.0.0.0:51001。
ROBOSERVER_LICENSE_LIMIT	RoboServer が受け取ることができるライセンス ユニットの最大数。
LOG4J_ROOTLOGGER	ルート ログ レベル。 デフォルト値: ERROR
LOG4J_LOGGER_WEBKIT	WebKit に関連するメッセージのログ レベル。 デフォルト値: INFO
LOG4J_LOGGER_KAPOW_SERVERMESSAGELOG	ロボットに関係しないメッセージのログ レベル。 デフォルト値: INFO
LOG4J_LOGGER_KAPOW_ROBOTRUNLOG	ロボットの実行メッセージのログ レベル。 デフォルト値: INFO
LOG4J_LOGGER_KAPOW_ROBOTMESSAGELOG	ロボットが記録するメッセージのログ レベル。 デフォルト値: INFO
LOG4J_LOGGER_KAPOW_AUDITLOG	各 HTTP/FTP リクエストのログ レベル。 デフォルト値: OFF
LOG4J_LOGGER_KAPOW_KCUMANAGER_KCUINFO	KCU 情報のログ レベル。 デフォルト値: OFF
LOG4J_LOGGER_HUB	Desktop Automation プロセスのログ レベル。 デフォルト値: WARN
LOG4J_APPENDER_FILE_MAXFILESIZE	ローテーション前のログ ファイルの最大サイズ。 デフォルト値: 10MB
LOG4J_APPENDER_FILE	ログ ファイルのパスとファイル名。 デフォルト値: /kapow/data/Logs/${logFileName}
LOG4J_APPENDER_FILE_MAXBACKUPINDEX	ローテーションされるログ ファイルの最大数。 デフォルト値: 3

Kapplets 変数

Kapplets の環境変数を設定するメカニズムは、Management Console の場合と同様です。

標準の設定プロパティとともに、コンテナ化された展開内の Kapplets 設定では次の環境変数を使用することができます。

環境変数	説明
JDBC_DRIVER_URL	JDBC ドライバーをダウンロードするための URL を指定します。
JAVA_OPTS	Java 仮想マシン (JVM) のオプションとパラメータを指定します。

環境変数	説明
SLEEP_DELAY	コンテナが起動する前の遅延を秒数で指定します。

ロボット ファイル システムの変数

ロボット ファイル システムの環境変数を設定するメカニズムは、Management Console の場合と同様です。

標準の設定プロパティとともに、コンテナ化された展開内のロボット ファイル システム設定では次の環境変数を使用することができます。

環境変数	説明
JAVA_OPTS	Java 仮想マシン (JVM) のオプションとパラメータを指定します。
SLEEP_DELAY	コンテナが起動する前の遅延を秒数で指定します。

Synchronizer 変数

環境変数	説明
SYNCHRONIZER_MC_URL	プロトコルとポート番号を含む、Management Console に接続するための URL。
SYNCHRONIZER_SHARED_SECRET	Management Console 用の Synchronizer 共有シークレット (プレーンテキスト)。
SYNCHRONIZER_SHARED_SECRET_FILE	Management Console の Synchronizer 共有シークレットを含むファイルへのパス。
SYNCHRONIZER_INTERVAL	同期を実行する間の秒単位の間隔。0、負の値、または数値以外の値に設定すると、Synchronizer は 1 度実行された後に終了します。
SYNCHRONIZER_PRIVATE_KEY	リモート リポジトリに接続するためのプライベート SSH キー ファイルへのパス。ローカル リポジトリの場合は無視されますが、値を指定する必要があります。
SYNCHRONIZER_NO_HOST_KEY	厳密な SSH ホスト キーのチェックを無効にします。 デフォルト値: false
SYNCHRONIZER_NO_GIT_WIPE	古いデータが検出された場合の Git データの消去を無効にします。 デフォルト値: false
SYNCHRONIZER_HEALTH_CHECK_TIMEOUT	Synchronizer に異常があると判断されるまでのタイムアウト (秒)。
LOG4J_LOGGER_ROOT_LOGGING	ルート ロガーのログ レベル。例: ERROR、WARN、INFO、DEBUG。
LOG4J_LOGGER_ADDITIONAL_KEY_<N>	追加の Log4j プロパティ番号 N のキー。

環境変数	説明
LOG4J_LOGGER_ADDITIONAL_VALUE_<N>	追加の Log4j プロパティ番号 N の値。

Docker Compose の例

Tungsten RPA には、Linux 環境向けのサンプル設定を含む複数の Docker 構成ファイルが用意されています。これらのファイルは以下の場所にあります。

[インストール_ディレクトリ]/docker/compose-examples

ファイル名	説明
docker-compose-basic.yml	Management Console、RoboServer、および PostgreSQL データベースを起動します。 Management Console の起動時に手動で入力する必要がないように、開始する前に、構成設定にライセンス情報を入力してください。
docker-compose-ha.yml	Management Console、RoboServer、PostgreSQL データベース、および軽量の Traefik イメージに基づくロード バランサーを起動します。 サービスを開始する前に、構成設定にライセンス情報を入力してください。カスタム hazelcast.yml ファイルを指定し、そのファイルをコンテナにマウントして、Management Console コンテナの SPRING_HAZELCAST_CONFIG 環境変数を設定します。
docker-compose-kapplets.yml	Management Console、RoboServer、PostgreSQL データベース、および Kapplets をその設定で起動します。
docker-compose-ldap.yml	LDAP 認証のデモを行います。テストの代わりとして、ldap_ad_content.ldif ファイルを含む OpenLDAP コンテナを使用します。本番環境では、既存の LDAP サーバーに接続してください。
docker-compose-rfs.yml	Management Console、RoboServer、PostgreSQL データベース、およびロボット ファイル システムをその設定で起動します。
docker-compose-synchronizer.yml	Management Console、RoboServer、PostgreSQL データベース、および Synchronizer をその設定で起動します。

Kubernetesサンプル Helm チャート

Tungsten RPA には、Kubernetes クラスタへの展開用のサンプル Helm チャートが用意されています。サンプルは以下の場所にあります。

[インストール_ディレクトリ]/helm.

このチャートを使用して、以下のコンポーネントを展開します。

- Management Console
- RoboServer (スケーラブル)

- Kapplets
- Synchronizer
- ロボット ファイル システム
- Management Console 用の PostgreSQL データベース
- Kapplets 用の PostgreSQL データベース
- Management Console、Kapplets、およびロボット ファイル システム用の Ingress リソース

デフォルトでは、このチャートにより、Management Console、RoboServer、および Management Console 用の PostgreSQL データベースで構成される最小限のセットアップが有効になります。

チャートを展開する前に、以下の前提条件を満たしていることを確認してください。

- 動作する Kubernetes クラスタ
- クラスタにインストールされたイングレス コントローラ
- (すべてが Kubernetes Ingress コントローラを指している) Management Console、Kapplets、およびロボット ファイル システムのエンドポイントの登録済み DNS 名。

チャートからファイルを抽出して独自の `values.yaml` ファイルを作成します。次のプロパティ グループを設定します。

- **enable**: 有効にするコンポーネント
- **license**: 使用するライセンス キー
- **roboserverReplicas**: 実行する RoboServer のインスタンス数
- **sharedSecret**: サービス認証に使用する共有シークレット
- **host**: コンポーネントにアクセス可能なホスト名。これらのホスト名を DNS プロバイダに登録して、適切に解決されるようにしてください。

サービスとしての RPA コンポーネントの実行

Tungsten RPA の各サーバー コンポーネントは Windows サービスとして実行することができるため、自動的に起動し、定義されたサービス アカウントで動作することが保証されます。

次のトピックでは、`ServiceInstaller.exe` を使用してコンポーネントをサービスとしてインストール、設定、および削除する方法について説明します。

- ServiceInstaller.exe を使用したインストール - パラメータとコマンド構文の概要。
- 管理コンポーネントをサービスとして実行 - Management Console、ロボット ファイル システム、Kapplets、および Synchronizer を Windows サービスとしてインストールして設定します。
- RoboServer をサービスとして実行 - RoboServer を Windows または Linux サービスとしてインストールして設定します。

ServiceInstaller.exe によるインストール

Tungsten RPA コンポーネントをサービスとして実行するためには、まず `ServiceInstaller.exe` プログラムを使用してそのコンポーネントをインストールする必要があります。以下

は、「RPAComponent」プログラムへのコマンドライン引数の概要を示す一般的な例です。(読みやすくなるように複数行で表示されていますが、1 行として実行されます。)

```
ServiceInstaller.exe -i RPAComponent.conf
wrapper.ntservice.account=Account
wrapper.ntservice.password.prompt=true
wrapper.ntservice.name=Service-name
wrapper.ntservice.starttype=Start-method
wrapper.syslog.loglevel=INFO
wrapper.app.parameter.1="First-Argument"
wrapper.app.parameter.2="Second-argument"
```

wrapper.ntservice.account

「RPAComponent」を実行する必要があるユーザーのアカウント。Tungsten RPA はユーザーのディレクトリに設定を保存するため、適切に設定されたユーザーを選択することが重要です。

「RPAComponent」をドメイン ユーザーとして実行する必要がある場合は、`domain\account` の形式でアカウントを入力します。

「RPAComponent」を通常のユーザーとして実行する必要がある場合は、`.\account` の形式でアカウントを入力します。

セキュリティ上の理由から、RoboServer サービスのログイン用の `LocalSystem` アカウントは使用しないでください。`LocalSystem` が使用されている場合、WebKit（デフォルト）ロボットの実行時に以下のエラーが発生します: 「WebKitBrowser への接続を確立できませんでした。バスへの接続に失敗しました。」

wrapper.ntservice.password.prompt

値が `true` の場合、アカウントパスワードの入力をユーザーに求めます。コマンド ラインにパスワードを入力する場合は、`wrapper.ntservice.password=<your-password>` を使用します。

wrapper.ntservice.name

インストールするサービスの名前。サービス名にスペースを含めることはできません。

wrapper.ntservice.starttype

次のいずれかの値を使用します。

- `AUTO_START`: システムの再起動時にサービスを自動的に開始します。
- `DELAY_START`: 少し時間を空けてサービスを開始します。
- `DEMAND_START`: サービスを手動で開始します。

wrapper.syslog.loglevel

「RPAComponent」からイベント ログへコンソール出力をリダイレクトします。

wrapper.app.parameter.

「RPAComponent」の引数。必要な数だけ入力してください。

サービスがインストールされると、「サービスとしてログオン」権限がユーザーに付与されます。サービスの開始に失敗した場合は、gpedit.msc を開いて権限が付与されていることを確認し、(Windows 10 の場合) **Administrative Tools** > **Local Security Policy** > **Local Policy** > **User Rights Assignment** > **Log on as a service** > **Properties** に移動して、ユーザーを追加します。

サービスとしての管理コンポーネントの実行

次の手順では、`ServiceInstaller.exe` を使用して、Tungsten RPA Management Console、ロボット ファイル システム、Kapplets、および Synchronizer を Windows サービスとしてインストール、設定、および削除する方法について説明します。

以下のスクリプトの例では、「RPAComponent」を、インストールするコンポーネントの適切な名前と設定ファイルで置き換えます。

コンポーネント	スクリプト内の名前	例
Management Console	ManagementConsole	ManagementConsole.conf 「ManagementConsole2026.1」
ロボット ファイル システム	RobotFileSystem	
Kapplets	Kapplets	
Synchronizer	シンクロナイザー	

Windows サービスの追加

1. コマンド プロンプト ウィンドウを開き、コマンド ラインからコンポーネントを起動するために必要なパラメータを指定します。

   詳細な起動パラメータについては、『Tungsten RPA のヘルプ』の関連トピックを参照してください。

   > Windows サービスを実行するアカウントと同じユーザー アカウントでコマンドを実行します。設定ファイルとアプリケーション ログは、コマンドを実行したユーザーの AppData フォルダに保存されます。

2. コンポーネントが正しく実行されていることを確認し、コマンド ラインで `-s` パラメータを使用して構成設定を保存します。
3. 次のスクリプトを使用して、コンポーネントをサービスとしてインストールします。(読みやすくなるように複数行で表示されていますが、1 行として実行されます。)

   ```
   ServiceInstaller.exe -i RPAComponent.conf
   wrapper.ntservice.account=domain\account
   wrapper.ntservice.password.prompt=true
   wrapper.ntservice.name="RPAComponent<VersionNo>"
   wrapper.ntservice.starttype=MANUAL
   wrapper.syslog.loglevel=INFO
   ```

4. 構成パラメータを変更するには、サービスを停止し、新しいパラメータを使用してコンポーネントを手動で実行して、設定を保存してから、サービスを再起動します。

Windows サービスの削除

サービスをアンインストールするには、次のコマンドを実行します。

```
ServiceInstaller.exe -r RPAComponent.conf
 wrapper.ntservice.name="RPAComponent<VersionNo>"
```

サービスとしての RoboServer の実行

RoboServer を Windows または Linux サービスとして設定します。

- Windows でのサービスとしての RoboServer の実行
- Linux でのサービスとしての RoboServer の実行

Windows でのサービスとしての RoboServer の実行

Windows で RoboServer が自動的に起動するように設定するには、`ServiceInstaller.exe` プログラムを使用して RoboServer を Windows サービスとして追加します。

Windows Server 上で動作する場合:

Windows Server 上で RoboServer をサービスとして実行する場合、プロセスは標準のユーザー ホーム ディレクトリではなく、`systemprofile` ディレクトリを使用します。サービス アカウントにはこの場所での権限がない可能性があります。

WebKit の起動エラーを防ぐには、次のフォルダを手動で作成します (必要に応じて、不足している親フォルダを作成します)。

- `C:\Windows\system32\config\systemprofile\AppData\Local\Tungsten RPA\[バージョン]\.dbus-keyrings\`
- `C:\Windows\SysWOW64\config\systemprofile\AppData\Local\Tungsten RPA\[バージョン]\.dbus-keyrings\`

[バージョン] を、インストールされている Tungsten RPA のバージョン番号に置き換えます。

Windows サービスの追加

次のスクリプトによって、デフォルトのパラメータを使用して RoboServer を起動するサービスをインストールします。サービスの名前は必要に応じて変更できます。(読みやすくなるように複数行で表示されていますが、1 行として実行されます。)

```
ServiceInstaller.exe -i RoboServer.conf
wrapper.ntservice.account="<DOMAIN>\<USERNAME>"
wrapper.ntservice.password.prompt=true
wrapper.ntservice.name="RoboServer2026.1"
wrapper.ntservice.starttype=AUTO_START
wrapper.syslog.loglevel=INFO
wrapper.app.parameter.1="-p"
wrapper.app.parameter.2="<PORT-NUMBER>"
wrapper.app.parameter.3="-mcUrl"
wrapper.app.parameter.4="<MC-URL>"
wrapper.app.parameter.5="-cl"
wrapper.app.parameter.6="<ROBOSERVER-CLUSTER>"
wrapper.app.parameter.7="-ss"
wrapper.app.parameter.8="<MC-SHARED-SECRET>"
```

Windows サービスの削除

サービスをアンインストールするには、次のコマンドを実行します。

```
ServiceInstaller.exe -r RoboServer.conf wrapper.ntservice.name=Service-name
```

Linux でのサービスとしての RoboServer の実行

RoboServer を `init.d` サービスとして実行するには、次の手順を実行します。

1. Linux パッケージをダウンロードし、デフォルトのディレクトリ `/opt/Tungsten RPA/` に抽出します。
2. DEB パッケージをインストールします。
3. RoboServer の設定を変更（オプション）
4. `#service RoboServer start` を実行するか、マシンを再起動してサービスを自動的に開始します。

   デフォルトでは、RoboServer は SSL ポートのリッスンを伴って起動します。`roboserver.settings` ファイルでポート番号を指定します。
5. 使用可能なコマンドを表示するには、引数なしで `/etc/init.d` から RoboServer を実行します。

Windows でのサイレント インストールとアンインストール

ユーザーが操作を行うことなく Windows への Tungsten RPA のインストールまたは削除を実行するには、次の手順を実行します。

- Windows でのサイレント インストール
- Windows でのサイレント アンインストール

Windows でのサイレント インストール

サイレント インストーラーは、ユーザーの操作なしで実行されます。これは、たとえばスクリプトでインストール プロセスを自動化する必要がある場合に便利です。

フル インストーラの使用

Tungsten RPA のサイレント インストールを実行するには、管理者権限で次のコマンドを実行します。

```
msiexec /qn /i TungstenRPA-2026.1.msi
```

このコマンドは、プログラムをデフォルトの場所にインストールします。別の場所を指定するには、次のコマンドを使用します。

```
msiexec /qn /i TungstenRPA-2026.1.msi INSTALLDIR="dir"
```

ここで、`"dir"` はインストールする場所です。例：

```
 msiexec /qn /i TungstenRPA-2026.1.msi INSTALLDIR="C:\Tungsten RPA 2026.1.0.0\"
```

⚠ Tungsten RPA をカスタム フォルダにインストールする場合、インストール フォルダとすべての親フォルダ名が、Windows インストールのシステム ロケールに準拠している必要があります。システム ロケール設定は、「非 Unicode プログラムの言語」の [地域の設定] にあります。

インストール プロセスを記録するファイルを指定するには、次のコマンドを使用します。

```
msiexec /qn /i TungstenRPA-2026.1.msi /l msilog.txt
```

限定インストーラの使用

以下は、サイレント モードでさまざまなインストーラーを使用する例です。

```
msiexec /qn /i TungstenRPADesignStudio-2026.1.msi
msiexec /qn /i TungstenRPARoboServer-2026.1.msi
msiexec /qb /i TungstenRPADesktopAutomationService-2026.1.msi
```

Windows でのサイレント アンインストール

Tungsten RPA のサイレント アンインストールを実行するには、管理者権限で次のコマンドを実行します。

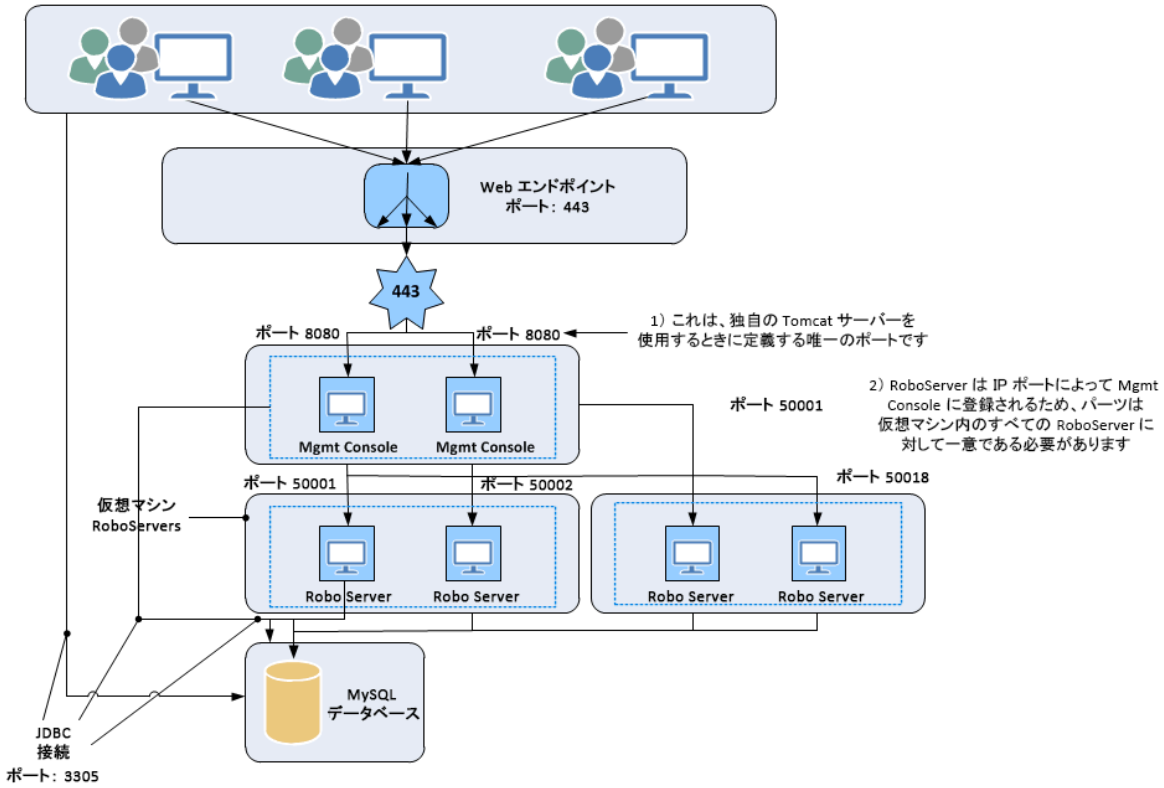
```
msiexec /qn /x TungstenRPA-2026.1.msi
```

アンインストール プロセスを記録するファイルを指定するには、次のパラメータを使用します。

```
msiexec /qn /x TungstenRPA-2026.1.msi /l*v uninstall.log
```

付録 A

Tungsten RPA セキュリティ モデル



A.ユーザーのログインと認証

カテゴリー	認証と承認
説明	Tungsten Automation アプリケーションのログイン クレデンシャルは、ユーザーが入力します。

セキュリティの詳細	Tungsten RPA はActive Directory / LDAP のユーザー/グループの同期をサポートします。これにより Tungsten RPA は認証および資格情報管理のために企業インフラストラクチャを活用します。 Tungsten RPA 利便性のために、アプリケーション固有の認証および承認メカニズムも備えています。これには資格情報管理とストレージが含まれます。保存されたパスワードは暗号化されます。

B.クライアントは Tungsten RPA サーバーに送信

カテゴリー	転送中のデータ
ポート	80 または 443
プロトコル	HTTP または HTTPS
説明	クライアントは Tungsten RPA サーバーに送信します。
セキュリティの詳細	Tungsten RPA クライアント（ Management Console と Design Studio ）から Tungsten RPA へのすべての接続は HTTP / HTTPS 経由です。HTTPS は、最高レベルのセキュリティを設定する必要があります。

C. Tungsten RPA サーバーが別の Tungsten RPA サーバーに送信

カテゴリー	転送中のデータ
ポート	設定可能。デフォルト 80、443、50000、50443、49999、49998
プロトコル	HTTP / HTTPS、ソケットTCP / IP
説明	Tungsten RPA サーバーは別の Tungsten Automation アプリケーションやサーバーとの間で送受信を行います。
セキュリティの詳細	すべての Tungsten RPA コンポーネントは、カスタム証明書のある安全な暗号化通信（TLS 1.2）を使用するように設定できます。

D. Tungsten RPA サーバーはデータベース サーバーに送信

カテゴリー	転送中のデータ
ポート	プロトコルによって異なります
プロトコル	TCP / IP
説明	Tungsten RPA サーバーがデータベースと送受信を行います。
セキュリティの詳細	Tungsten RPA サーバーは SQL データベースに接続します。 通常、データベース サーバー システムは同じ場所に配置されるか物理的に保護されるため、送信を暗号化する必要はありません。 ただし、このような暗号化が必要な場合は、SSL を介してデータベース接続を暗号化できます。

E.ロボットとデータ ストレージ

カテゴリー	REST のデータ
説明	ロボット、設定、および関連するメタデータは、Management Console を介して保存されます。ロボットは顧客データをデータベースに保存できます。
セキュリティの詳細	ロボット、設定、および関連するメタデータは、Tungsten Automation データベースに保存されます。これらのメタデータには、設定されたシステム アカウントを介してアクセスします。データベース自体の暗号化機能を使用して、データベース レベルの暗号化も利用できます。 ファイル システムやデータベースの暗号化が有効であるかどうかに関係なく、パスワード (外部システム用またはアプリケーション固有のユーザー用) がさらに保護されます。[Vault] に保存されたパスワード、またはスケジュールへの入力として保存されたパスワードは、顧客が生成した証明書を使用して暗号化されます。証明書に選択された暗号は、保存されているパスワードを暗号化するために使用されます。デフォルトでは、インストールには RSA 1024 ビット暗号化証明書が設定されていますが、顧客が証明書を生成するようにすることを強くお勧めします。詳細については、「Management Console のデータ暗号化」を参照してください。