# 1Integrate

# Installation Guide

## WildFly Application Server

1Spatial
Tennyson House
Cambridge Business Park
Cambridge
CB4 0WZ
United Kingdom

**Phone:** +44 (0)1223 420414

**Fax:** +44 (0)1223 420044

**Web:** www.1spatial.com

# Contents

# 1 Introduction

This guide explains how to install 1Integrate.

The procedures apply to both Windows and Linux environments, unless specifically indicated.

For more information about new features and changes in this release, and hardware and software requirements, refer to the *1Integrate Release Notes*.

## Audience

This guide is intended for personnel responsible for the installation, configuration, and administration of software.

The procedures detailed in the guide should be performed by a system administrator who is familiar with the application environment of the organisation.

## Licences

1Integrate licences will be issued via email.

## 1Spatial Product Support

If assistance is required while installing 1Integrate, please call 1Spatial support on +44 (0)1223 423069, or visit the support section of the 1Spatial website via the Services menu at www.1spatial.com.

# 2 Prerequisites

Before installing 1Integrate, please ensure you have met all system requirements and installed all necessary prerequisite components:

- ▶ Check "System Requirements" below (including the version of Java required)
- ▶ "Microsoft Visual C++ Redistributable (Windows only)" on the next page
- ▶ "Configuring a Database Server for 1Integrate's repository" on the next page
- ▶ "ICU Libraries (Linux only)" on page 9
- ▶ "Required Packages (Linux only)" on page 8
- ▶ "Loading Data Formats" on page 26

## System Requirements

Please see the *1Integrate Release Notes* for specific system requirements for this release.

## Sizing a Server for 1Integrate

There are a number of minimum requirements for server size in order to install 1Integrate.

Depending on your intended configuration you will need, as a minimum:

per Interface:

- ▶ 2 CPU Cores
- ▶ 2GB RAM

per Engine:

- ▶ 1 CPU Core
- ▶ 1GB RAM

> **Note:** In order to achieve optimal performance, a dedicated core per engine is recommended.

While being processed, data is stored on disk. So the recommended available storage varies on a case by case basis.

While improving CPU and memory will increase performance, the biggest gains can be made from using fast disk storage, e.g. SSDs.

> **Note:** For storage requirements, a minimum of 10GB should cover basic uses of your 1Integrate installation. Additional storage will be required for the database you will be using.

# Microsoft Visual C++ Redistributable (Windows only)

Microsoft Visual C++ 2015-2019 64-bit Redistributable packages are required for Windows installations of 1Integrate.

These can be downloaded from the Microsoft website (**vcredist_x64.exe**).

For more information about the version numbers of pre-requisites, please refer to the relevant release notes for your installation.

# Configuring a Database Server for 1Integrate's repository

1Integrate requires access to an database server to store configuration data such as rule or session definitions, and conformance results. This is referred to as the "1Integrate repository".

Supported Database Servers are:

▶ *Oracle* Enterprise

▶ *PostgreSQL (WildFly only)*

▶ *Microsoft SQL Server (WildFly only)*

> **Note:** Please refer to the relevant product documentation when installing your chosen database server.

## Creating the repository

You will need to create a database and user (Oracle only requires a User) for your 1Integrate repository on your selected server.

> **Note:** When deploying 1Integrate to multiple servers, one database user is required per environment.

Oracle:

Using Oracle, 1Integrate requires a database user. The setup should be performed as recommended by your Database Administrator who must grant the user access to the database and at least the following permissions:

- ► `create sequence`
- ► `create table`

Microsoft SQL Server Database :

Using SQL Server Database use, you will need a Database and a User that can access that database.

The setup should be performed as recommended by your Database Administrator who must grant the user access to the database and at least the following permissions:

- ► `create sequence`
- ► `create table`

> **Note:**
> Microsoft SQL server supports two methods of authenticating users, but these will differ depending on your operating system:
>
> - ► Windows: SQL Server authentication and Integrated Windows Authentication
> - ► Linux: SQL Server authentication

Create a PostgreSQL User :

Create a user that can access that database with the default properties. This will ensure the user has all required privileges. Alternatively the set up should be performed as recommended by your Database Administrator who must grant the user access to the database and at least the following permissions:

- ► `create sequence`
- ► `create table`

# Required Packages (Linux only)

The following package is a prerequisite for Linux installations of 1Integrate:

- ► **libX11**

You must install **libX11** via the system's package manager.

# ICU Libraries (Linux only)

ICU libraries are required for data and timestamp support in Linux.

A root user (or a user with root access privileges) is required to copy the files from the installation package and run the `ld_config` command.

Install the ICU libraries:

1. Run the `su` command to switch to the root user.

2. Copy the **.so** files from the installation folder (within the ICU folder) to the **/usr/local/lib64/** folder.

3. Create a new configuration file: **/etc/ld.so.conf.d/integrate.conf**

> **Note:** Both the **.so** files and the **.conf** file must have read permissions for all users.

4. Inside the configuration file, reference the location of the ICU libraries, for example: `/usr/local/lib64/*`

5. Enter the following `ld_config` command:

```
/sbin/ldconfig -v /usr/local/lib64/
```

Alternatively, edit **~/.bashrc** or similar for the user used to run 1Integrate to include `/usr/local/lib64` on the `LD_LIBRARY_PATH`, as in the following example:

```
# User specific
if [ -z "$LD_LIBRARY_PATH" ]; then
export LD_LIBRARY_PATH="/usr/local/lib64"
else
export LD_LIBRARY_PATH="/usr/local/lib64:$LD_
LIBRARY_PATH"
fi
```

# Data Store Prerequisites

1Integrate supports the following types of data stores for input (reading) and output (writing), either as standard or using FME Desktop (which requires a licence). Pay particular attention to the prerequisites attached to certain data store types:

> **Note:** All formats listed below as using FME Desktop require FME Desktop Professional Edition, unless otherwise indicated.

---

| Data Store Type | As standard | | Using FME Desktop | |
|---|---|---|---|---|
| | Read | Write | Read | Write |
| Autodesk AutoCAD DWG/DXF | ✗ | ✗ | ✓ | ✓ |
| Bentley Microstation Design (V8) | ✗ | ✗ | ✓ | ✓ |
| Delimiter-separated value (DSV) | ✓ | ✓ | ✗ | ✗ |
| Esri ArcGIS Server | ✓ | ✓ | ✗ | ✗ |
| Esri Enterprise Geodatabase | ✗ | ✗ | ✓[1] | ✓[2] |
| Esri File Geodatabase | ✓ | ✗ | ✓ | ✓ |
| Esri Shapefile | ✓ | ✓ | ✓ | ✓ |
| Google BigQuery | ✓ | ✓ | ✗ | ✗ |
| MapInfo Tab | ✓ | ✓ | ✓ | ✓ |
| Microsoft SQL Server Spatial | ✗ | ✗ | ✓[3] | ✓[4] |
| Oracle | ✓ | ✓ | ✗ | ✗ |
| PostGIS | ✓ | ✓ | ✗ | ✗ |

[1]This format is only available on Windows. Requires FME Desktop (64-bit) Esri Edition, and ArcGIS Desktop, ArcGIS Desktop Background Processing (64-bit) and Database client (64-bit).

[2]This format is only available on Windows. Requires FME Desktop (64-bit) Esri Edition, and ArcGIS Desktop, ArcGIS Desktop Background Processing (64-bit) and Database client (64-bit).

[3]This format is only available on Windows. The Microsoft SQL Server 2012 Native Client must be installed.

[4]Windows only. Requires FME Desktop Database Edition. The Microsoft SQL Server 2012 Native Client must be installed.

| Data Store Type | As standard | | Using FME Desktop | |
| --- | --- | --- | --- | --- |
| | Read | Write | Read | Write |
| OGC Web Feature Service | ✓[1] | ✗ | ✗ | ✗ |

[1]This format only supports 2D geometries.

# 3 Installing 1Integrate on WildFly

> **Note:** Before proceeding, ensure you have completed all pre-requisite steps (see "Prerequisites" on page 6).

1Integrate runs as several separate applications: an interface application and one or more engine applications.

The following instructions describe how to configure an interface and an engine. This is suitable for running both parts of 1Integrate on one host.

## Deployment Preparation

The following information is required prior to deploying 1Integrate:

▶ JDBC URL of the repository created

▶ Username and password for the database user

▶ Path of the **tnsnames.ora** file (if used)

▶ Location of a directory that can be used to store 1Integrate temporary files

> **Note:** On both Windows and Linux, paths must be entered with forward slashes (/) as the path separator instead of backslashes (\). Ensure that no spaces are entered after each parameter as this will result in an unsuccessful installation.

## Copying the WildFly directory

Copy the WildFly directory (typically called `1Integrate-[version]-wildfly`) from the installation package to a local folder, then extract the files to a location such as **C:\Program Files\1Spatial\1Integrate\**. These files will be used to configure settings for 1Integrate in the following procedures.

In the WildFly installation, the **standalone** directory contains the configuration for the 1Integrate interface and the **standalone-engine** directory contains the configuration for the engine.

# Configuring System Properties

The following parameters can be configured in the **settings.properties** file, located within the **SETTINGS** folder within the installation location (e.g. C:\Program Files\1Spatial\1Integrate\1Integrate-[version]-wildfly\SETTINGS).

> **Warning:** References to `sessionqueue` are being updated to `engine`. When copying the **settings.properties** file to a new installation during an upgrade, ensure these references are updated. For more information, please see the **Release Notes** within your installation package.

> **Note:** On Windows, folder paths within the **settings.properties** file must be specified using a forward slash. Spaces in folder names are allowed (e.g. C:/Program Files/1Spatial/1Integrate).

| Parameter | Description |
|---|---|
| **Repository Database Connection Details** | |
| repository.driver | The database type (oracle, sqlserver or postgresql). |

| Parameter | Description |
|---|---|
| repository.url | The URL for the database in which to store 1Integrate metadata and entities (rules, actions, sessions and so on), and files that are uploaded to the server used by the data store (such as MapInfo Tab files). |

This is in the format:

▶ for Oracle: `jdbc:oracle:thin:@[HOSTNAME]:[PORT]/[Service]`

Or

`jdbc:oracle:thin:@[HOSTNAME]:[PORT]:[SID]`

▶ for MS SQL: `jdbc:sqlserver://[serverName[\\instanceName][:portNumber]];databaseName=1Integrate`

for MS SQL with Windows Authentication: `repository.url=jdbc:sqlserver://[serverName[\\instanceName][:portNumber]];integratedSecurity=true;databaseName=1Integrate`

> **Note:** You must define either a `serverName` or `instanceName`  See the Microsoft documentation for further guidance.

▶ for PostgreSQL:

> **Note:** You will only need to specify a port if you are not using the default.

▶ To use the *public schema* to store the repository:

```
jdbc:postgresql://[HOSTNAME]:[PORT(if non-default)]/[DATABASE]
```

▶ To use a *specified schema* to store the repository:

```
jdbc:postgresql://[HOSTNAME]:[PORT(if non-default)]/[DATABASE]?currentSchema=[REPOSITORYSCHEMA]
```

| Parameter | Description |
|---|---|
| repository. username | Username to connect to the repository schema, in which to store 1Integrate metadata and entities (e.g. rules, actions, sessions) as well as files that are uploaded to the server used by the data store (e.g. MapInfo Tab files). |
| | **Note:** Ensure the selected user has the required privileges to access the database. |
| repository. password | Password for the user described by repository.username. |
| repository.valid-sql | Add the validation parameter to check for database restart. |
| | **For SQL Server or PostgreSQL:** |
| | repository.valid-sql=select 1 |
| | **For Oracle:** |
| | repository.valid-sql=select 1 from dual |
| | **Note:** Further changes are required in the **standalone.xml** to auto reconnect to restarted databases. |
| **Licence file Location** | |
| ls_license | The location and name of the provided product licence file. |
| | **Note:** The licence file is provided by 1Spatial Support and can be placed in any folder. |

| Parameter | Description |
|---|---|
| **Location for Temporary Files** | |
| temp.directory | This is a temporary folder location for files that are eventually stored in the database. |
| | For Linux, it is recommended that this is set to `/tmp` or `/var/tmp`. |
| | For Windows, it is recommended that this is set to `C:/WINDOWS/TEMP`. |
| | **Note:** It is recommended that the path for your temp.directory is concise, otherwise you may encounter issues (such as engine crashes or session errors) with specific data formats. |
| **Location for the Cache Directory** | |
| cache.directory | The location of the cache directory. |
| | When a session is run, a folder is created called "1Integratecache", within which the cache is stored. |
| | **Note:** This stores the data cache from data read by 1Integrate. This may require large amounts of disk space depending on the size of data being read into sessions and the number of concurrent sessions. |
| | For Linux, if left blank this will default to `/tmp`. |
| | For Windows, if left blank this will default to `C:\Users\[user]\AppData\Local\Temp`. |
| **WildFly Server Port Settings** | |
| interface.http.port | The port to use for the interface. |
| | **Note:** If possible, use the default port number specified in the file. If there is a clash with another application that already uses this port number, increment the value by 1 until an unused port number is found. |

| Parameter | Description |
|---|---|
| engine.http.port | The port to use for the engine.<br><br>**Note:** If possible, use the default port number specified in the file. If there is a clash with another application that already uses this port number, increment the value by 1 until an unused port number is found. |
| **Additional Settings** | |
| studiotask.countTotalObjects | If set to true, a progress bar is displayed in the user interface during open data tasks to allow the user to see what proportion of the total objects have been loaded so far. This reduces performance for an open data task.<br><br>If set to false, only the number of objects currently loaded is shown and the user is not shown the progress. This improves performance for an open data task, particularly when loading data from non-database sources.<br><br>**Note:** Counting the total number of target objects in order to calculate the progress may take excessive time to complete. It is recommended that this parameter is set to **false** for production deployment or when running sessions via web services. |
| grid.local.address<br><br>grid.communication.tcp.port | (optional) See "NIC/Network Adaptor Configuration" on the facing page. |

# Configure Login Banner

Optionally, you can configure a login banner to appear at the top of the Login page:

1. First, you must create a valid HTML file with plain text and optional heading tags. Text will be displayed as black.

2. In the **settings.properties** file, add the following:

| Parameter | Description |
|---|---|
| loginBanner | Enter a file path to a HTML document you have defined e.g: `loginBanner=[Location]/[File].html` |

> **Note:** The HTML page must be placed on the same server as the one running the Interface(s).

# NIC/Network Adaptor Configuration

The Grid discovery used to find engines by default uses the first found non-loopback address, for example a machine with Ethernet adaptors "eth0" and "eth1" and Local Loopback "lo" will likely use "eth0".

> **Note:** If you do not need to override the adaptor default behaviour, then the following properties do not need to be included.

## Configure NIC/Network Adaptor :

The following properties must be included in the **settings.properties** file:

```
grid.local.address=[NIC Address]
grid.discovery.tcp.port=[default: 51300]
grid.communication.tcp.port=[default: 51401]
```

Where:

- **grid.local.address** - specifies the IP address of the network adaptor used for grid communication.

- **grid.communication.tcp.port** and **grid.discovery.tcp.port** - allows environments to specify known ports (for example, when using a firewall).

> **Note:** The communication port must be a minimum of 100 greater than the discovery port, in order to avoid conflict.

# Reconnect Repository on database restart

It is possible to configure your WildFly deployment of 1Integrate to reconnect to repositories on a database restart.

> **Note:** You will need to make additions to the **settings.properties** file (for more, see the Configuring System Properties table and the `repository.valid-sql` parameter.

Once you have configured the **settings.properties** file you will also need to update the two versions of the **standalone.xml** in your **standalone** and **standalone-engine** folders of the1Integrate directory.

Place the following inside the data source validation tags for `jdbc/Repository` and `jdbc/RepositoryLocal`:

```
<check-valid-connection-sql>${repository.valid-
sql}</check-valid-connection-sql>

<validate-on-match>false</validate-on-match>

<background-validation>true</background-validation>

<background-validation-millis>10000</background-
validation-millis>
```

# Configure Memory in WildFly

To update the maximum available heap size in a 1Integrate WildFly installation, you can change the start scripts or set an environment variable.

## Setting the environment variables

You can set `INTEGRATE_ENGINE_OPTS` and `INTEGRATE_INTERFACE_OPTS` as environment variables.

If you do this, settings will persist between installations as you will not need to edit the start-up scripts.

## Changing the start-up scripts

Windows:

1. Edit **start-Interface.cmd** or **start-Engine.cmd**.

2. Locate the following lines:

```
SET "INTEGRATE_INTERFACE_OPTS=%INTEGRATE_
INTERFACE_OPTS%"
```

or

```
SET "INTEGRATE_ENGINE_OPTS=%INTEGRATE_ENGINE_
OPTS%"
```

3. Define the required size using a Java system property, e.g. to set the interface to 2GB:

```
SET "INTEGRATE_INTERFACE_OPTS=%INTEGRATE_
INTERFACE_OPTS% -Xms2g -Xmx2g"
```

> **Note:** For performance reasons it is recommended that your initial heap size matches your maximum heap size.

Linux:

1. Edit **start-Interface.sh** or **start-Engine.sh**.

2. Locate the following lines:

```
export "INTEGRATE_INTERFACE_OPTS=%INTEGRATE_
INTERFACE_OPTS%"
```

or

```
export "INTEGRATE_ENGINE_OPTS=%INTEGRATE_ENGINE_
OPTS%"
```

3. Define the required size using a Java system property, e.g. to set the interface to 2GB:

```
export "INTEGRATE_INTERFACE_OPTS=%INTEGRATE_
INTERFACE_OPTS% -Xms2g -Xmx2g"
```

> **Note:** For performance reasons it is recommended that your initial heap size matches your maximum heap size.

# Configuring Users and Roles

Users and Roles can be edited within 1Integrate.

> **Note:** You will need to restart 1Integrate for any changes to user and roles to take effect.

By default, 1Integrate is deployed with example users and passwords included. This enables a quick set-up process, but for security reasons it is HIGHLY RECOMMENDED that:

▶ As a minimum, on installation, change all passwords from the default to unique values.

▶ change the user names to ones relevant to your organisation.

For stronger security and management, consider using other authentication mechanisms such as using your organisation's Lightweight Directory Access

Protocol (LDAP) Service e.g. Microsoft Active Directory. This ensures that passwords and usernames are not stored in the application server but managed, as normal, by an IT department.

## Roles

The following roles are available in 1Integrate:

| Role | Description |
|---|---|
| rs_admins | The administrator can set up system parameters and has all the privileges of the other roles.<br><br>These include creating and modifying:<br><br>▶ rules<br>▶ data stores<br>▶ sessions<br>▶ actions<br>▶ action maps<br><br>The administrator can also define sessions and upload files. |
| rs_data_ engineers | A data engineer can:<br><br>▶ create and modify actions<br>▶ create and modify action maps<br>▶ define and run sessions |
| rs_data_ loaders | A data loader can upload files into 1Integrate on the Data Stores page only. |
| rs_data_ quality_ stewards | A data quality steward can define and run sessions only. |
| rs_rule_ definers | A rule definer can create and modify:<br><br>▶ rules<br>▶ data stores<br>▶ sessions<br><br>The rule definer can also run sessions. |
| rs_users | A user can only view data presented on the interface. |
| rswsuser | A web service user can use the web services. |

# Users

The following users are and roles are created by default upon installation:

| Username | Password | Assigned roles |
|---|---|---|
| INTFull | integrate1 | rs_users<br>rswsuser<br>rs_rule_definers<br>rs_data_loaders<br>rs_data_engineers<br>rs_data_quality_stewards |
| INTAdmin | integrate101 | rs_admins<br>rs_users<br>rswsuser |
| INTUser | integrate102 | rs_users |
| INTWService | integrate103 | rswsuser |
| INTRule | integrate104 | rs_rule_definers |
| INTLoader | integrate105 | rs_data_loaders |
| INTEng | integrate106 | rs_data_engineers |
| INTQuality | integrate107 | rs_data_quality_stewards |

# WildFly Users

To configure users and roles, navigate to the **\wildfly-[version]\SETTINGS** folder. This folder contains the following files:

▶ **users.properties** contains a list of usernames and passwords, in the form `username=password`.

> **Note:** All users listed in the following table are included as default.

▶ **roles.properties** contains a mapping from user names to 1Integrate roles in the form `username=role1,role2,role3`

# LDAP

For stronger security and management, Consider using other authentication and authorisation mechanisms such as your organisation's Lightweight Directory Access Protocol (LDAP) Service e.g. Microsoft Active Directory. This

ensures that passwords and usernames are not stored in the application server but managed, as normal, by an IT department.

# Authenticate using LDAP

The default WildFly configuration of storing passwords as plain text is not a recommended for production use. To configure 1Integrate to use your organisation's LDAP service in WildFly, perform the following configuration:

Configure an LDAP service:

1. Find the `settings.properties` file, to locate this go to: [1Integrate Directory]\SETTINGS\.

2. Uncomment the following and fill with your LDAP values:

```
#ldap.authentication.enabled=true
#ldap.host=
#ldap.principal=
#ldap.credential=
#ldap.username.attribute=
#ldap.user.base.dn=
#ldap.group.attribute=
#ldap.group.base.dn=
```

| Parameter | Expected Value |
|---|---|
| ldap.host | The hostname for the LDAP server. |
| ldap.principal | The principal (username) used for authentication with the LDAP server. |
| ldap.credential | The credential (password) used for authentication with the LDAP server. |
| ldap.username.attribute | The LDAP attribute to be used for 1Integrate login names ie.g. samAccountName |
| ldap.user.base.dn | The LDAP search root for users. |

| Parameter | Expected Value |
| --- | --- |
| `ldap.group.attribute` | The group name attribute to be used from your LDAP server e.g. samAccountName |
| `ldap.group.base.dn` | The LDAP search root for groups. . |

## Change default ldap settings:

Depending on your particular LDAP implementation, you may need to change some of the default LDAP settings.

```
# Enable if you need to override any defaults
#ldap.protocol=ldap
#ldap.port=389
#ldap.referral.mode=IGNORE
#ldap.direct.verification=true
#ldap.recursive.search=true
#ldap.group.member.attribute=member
#ldap.authentication.level=simple
```

To do this:

1. Find the `settings.properties` file, to locate this go to: [1Integrate Directory]\SETTINGS\.

2. Uncomment the sections that need changing for your active directory.

| Parameter | Expected Value |
| --- | --- |
| `ldap.protocol` | The protocol the connection will use (ldap or ldaps) |
| `ldap.port` | The port number for your LDAP |
| `ldap.referral.mode` | Set to FOLLOW if you have multiple LDAP servers that refer to eachother. |

| Parameter | Expected Value |
|---|---|
| `ldap.direct.verification` | Set to FALSE if WildFly should query the users password out of the LDAP server for user credential verification. <br><br> **Note:** For most installations direct verification will be sufficient. |
| `ldap.recursive.search` | Recursive search will search through nested groups. If this setting is unnecessary for your environment, setting to `false` may improve performance. |
| `ldap.group.member.attribute` | The attribute on the group object that contains the members of that group. |
| `ldap.authentication.level` | Set to none if you want to bind to the LDAP server anonymously. <br><br> **Note:** For security we would recommend all connections are authenticated. |

Mapping LDAP groups to 1Integrate roles:

Once you have completed your LDAP configuration, any users assigned to LDAP groups named the same as 1Integrate roles (e.g. `rs_admins`) will be authenticated and authorised by your LDAP system.

Authorisation in Wildfly:

It is possible to configure authentication so that usernames and passwords are managed by LDAP and the authorisation (roles assigned to specific LDAP groups) is managed within the configuration files. This removes the need for the LDAP service to know anything about 1Integrate specific roles.

> **Note:** In order authorise in LDAP and authenticate in WildFly, the `roles.properties` file must be populated with the **groups** that match those from the LDAP directory with the associated role(s). e.g. `LDAPGroup = rs_admins`.

> **Note:** You can only map **LDAP Groups** to roles if using this method.

Example of the `roles.properties` file which is mapping LDAP groups to the relevant roles:

```
ADMIN\ USERS= rs_admins,rs_users,rswsuser,rs_rule_
definers,rs_data_loaders,rs_data_engineers
ENGINEERS=rs_data_engineers
WORKERS=rs_users,rs_rule_definers
```

# Loading Data Formats

1Integrate can handle spatial data from a number of sources such as an Oracle database, Esri Shapefiles or MapInfo Tab files.

> **Note:** For more details on the formats supported, please see the
> [1Integrate online help](#).

## Using FME

To access other formats such as DWG files or Esri Enterprise geodatabases, 1Integrate uses functionality from Safe Software's FME. To use this capability, you must have FME Desktop installed and available to 1Integrate, with an FME Desktop license available for use.

> **Note:** In order to be editable, tables must have a primary key defined. You can read data from tables that do not have primary keys, but you will not be able to write data back to those tables.

To allow 1Integrate to access data that is only available via an FME license, use the following procedure, depending on your operating system:

## Configure FME on Windows

> **Note:** The following steps are only necessary if FME has not already been added to the system or user's PATH environment variable in Windows. However, when using WildFly with AdoptOpenJDK ensure the `plugins` path is defined, as detailed below.

On Windows, set the FME_HOME directory to the path in **bin\standalone.conf.bat** by adding the following lines to the top of the file:

> **Note:** The .bat file is located in the 1Integrate directory e.g.
> `\1Integrate[version]\wildfly[version]\bin`

**Using AdoptOpenJDK**

> **Note:** As of 1Integrate 2.7, 1Integrate only supports AdoptOpenJDK on WildFly installations. Please see the Release Notes for additional information.

```
@rem add FME to PATH
set PATH=[FME path];[FME path]/plugins;%PATH%
```

Where `[FME path]` is the location where FME is installed (e.g. C:\PROGRA~1\FME).

## Configure FME on Linux

Set the FME_CORE directory to the LD_LIBRARY_PATH in **bin/standalone.conf** by adding the following lines to the top of the file:

```
# add FME to LD_LIBRARY_PATH
export LD_LIBRARY_PATH="[FME_CORE_PATH]:$LD_LIBRARY_
PATH"
```

Where `[FME_CORE_PATH]` is the location of the fme/fmecore/ directory.

# Offline Help

By default, clicking the Help button within 1Integrate opens the online WebHelp. However, this button can be configured to open local offline documentation instead.

> **Note:** The configuration of offline help is only advised for environments without access to the internet, as local help files will not receive updates as often as the online help.

Before performing the configuration, ensure your local help files are places somewhere accessible on your local network. If the help files have not been provided with your release, please contact your vendor or 1Spatial Support.

Configure Offline Help :

1. Open the **settings.properties** file (located within C:\Program Files\1Spatial\1Integrate\1Integrate-[version]_wildfly\SETTINGS).

2.  Edit the value for the **help_base_url** parameter to the location of your local help files.

    ```
    help_base_url=[location of help files]
    ```

    Where `[location of help files]` is replaced with the path to your local help files.

# 4 Deployment

Once you have installed 1Integrate, it is now ready to be deployed. The method of deployment will differ depending on your Operating system or deployment type.

## Standard deployment

Before running your start-up scripts, you must first ensure your deployment is configured to refer to the correct version of Java. For the Java version supported for your release of 1Integrate, please refer to the release notes.

### JAVA_HOME

Once you are sure you have the correct version of Java installed, you will need to ensure that the JAVA_HOME variable is defined.

> **Note:** If the JAVA_HOME variable is already defined on your system with the correct version of Java, then you do not need to do anything.

If the variable is not set, first check that the default option in the **start-Interface** and **start-Engine** point to the correct location:

**Windows**

```
SET "DEFAULT_JAVA_HOME=C:\PROGRA~1\Java\jdk[version_
number]"
```

**Linux**

```
export DEFAULT_JAVA_HOME="/usr/java/jdk[version_
number]"
```

If the default is the wrong path to your Java, then carry out the following steps:

Windows:

1. Edit **start-Interface.cmd** and **start-Engine.cmd**.

2. Locate the following lines:

   ```
   @REM Uncomment and set this path to be correct
   for your environment
   REM SET "JAVA_HOME=C:\PROGRA~1\Java\[jdk_
   version]"
   ```

3. Uncomment the second line and remove "REM", so that it begins with "SET". Ensure that it is pointing to the location of Java home for the correct version of Java installed as part of the pre-requisites. The resulting line should appear as follows:

```
SET "JAVA_HOME=C:\PROGRA~1\Java\[jdk_version]"
```

4. Once configured, open the files to start the 1Integrate components.

### Linux:

1. Edit **start-Interface.sh** and **start-Engine.sh**.

2. Uncomment the line just below "`#Uncomment and set this path to be correct for your environment`":

```
export JAVA_HOME="/usr/java/[jdk_version]"
```

3. Once configured, open the files to start the 1Integrate components.

## Deploying the installation

You can now run the start-up scripts:

▶ **start-engine**

▶ **start-interface**

Once run, 1Integrate will be ready to use. For information on ensuring your installation and deployment has been successful, please refer to Testing the Installation.

## Configure additional Engines

1Integrate can be deployed with multiple engines each of which can run either on the same server as the interface or on separate machines (as long as they are on the same network as the interface).

Engines can be run in parallel, deployed using different port numbers. This requires a license from 1Spatial that enables the required number of parallel engines.

> **Note:** On a machine running Windows, this may present a path name length error. If this occurs, copy the **standalone-engine** folder to a higher level directory, rename it, and paste it back into the **wildfly-[version]** folder.

> **Note:** For more advanced environments, contact either your distributor or 1Spatial prior to continuing to discuss an appropriate deployment plan.

You will need to copy and edit a different file, depending on if you are using Windows or Linux.

Windows:

1. Create a copy of the **standalone-engine** folder and rename it **standalone-engine2** etc.

   > **Note:** You must suffix standalone-engine with an integer.

2. Inside the cloned directory, open the **configuration\engine.conf.bat**

3. Update `SET "ENGINE_NUMBER=1"` to the corresponding number e.g `SET "ENGINE_NUMBER=2"`

4. Create a copy of the **start-Engine.cmd** file and rename it.

   > **Note:** When deploying multiple engines, ensure you run each **start-Engine.cmd** file you have created.

5. Edit this file to refer to `"ENGINE_NUMBER=2"` instead of `"ENGINE_NUMBER=1"` (or `"ENGINE_NUMBER=3"` for the third engine etc.)

Linux:

1. Create a copy of the **standalone-engine** folder and rename it **standalone-engine2**.

   > **Note:** You must suffix standalone-engine with an integer.

2. Create a copy of the **start-Engine.sh** file and rename it **start-Engine2.sh**.

   Edit this file to refer to **standalone-engine2** instead of **standalone-engine** (or standalone-engine3 for the third engine etc.) and change the port offset at the end of the WildFly startup line from 100 to 200 (or 300 for the third engine etc.).

> **Note:** For example, the line
> ```
> ./bin/standalone.sh -
> P=SETTINGS/settings.properties -
> Djboss.server.base.dir=standalone-engine
> =Djboss.socket.binding.port-offet=100
> ```
> would be changed to
> ```
> ./bin/standalone.sh -
> P=SETTINGS/settings.properties -
> Djboss.server.base.dir=standalone-engine2
> =Djboss.socket.binding.port-offet=200.
> ```

## Engine Labels

You can label Engines you are deploying to group them, and control which sessions run on different groups.

### Windows:

1. Open the **standalone-engine** directory for the engine you will be applying labels to.

2. Within the **configuration** directory, open the **engine.conf.bat** file.

3. Locate and uncomment the following lines:

   ```
   SET "engine.labels="
   ```

4. Add engine labels as a comma separated list:

   ```
   SET "engine.labels=label1,label2,label3"
   ```

5. Repeat the steps for each engine being deployed.

### Linux:

1. Open the **start-Engine.sh** script for the engine you are applying labels to.

2. Locate and uncomment the following lines:

   ```
   export engine.labels=""
   ```

3. Add engine labels as a comma separated list:

   ```
   export engine.labels="label1,label2,label3"
   ```

4. Repeat the steps for each engine being deployed.

# Service Deployment (Windows Only)

You can deploy 1Integrate as a service through additional configuration and by running two scripts supplied in the installation package.

# JAVA_HOME

Once you are sure you have the correct version of Java installed, you will need to ensure that the JAVA_HOME variable is defined.

> **Note:** If the JAVA_HOME variable is already defined on your system with the correct version of Java, then you do not need to do anything.

If the variable is not set, first check that the default option in the **start-Interface-Service.cmd** and **start-Engine-Service.cmd** point to the correct location:

```
SET "DEFAULT_JAVA_HOME=C:\PROGRA~1\Java\jdk[version_
number]"
```

If the default is the wrong path to your Java, then carry out the following steps:

1. Edit **start-Interface-Service.cmd** and **install-Engine-Service.cmd**.

2. Locate the following lines:

```
@REM Uncomment and set this path to be correct
for your environment
REM SET "JAVA_HOME=C:\PROGRA~1\Java\[jdk_
version]"
```

3. Uncomment the second line and remove "REM", so that it begins with "SET". Ensure that it is pointing to the location of Java home for the correct version of Java installed as part of the pre-requisites. The resulting line should appear as follows:

```
SET "JAVA_HOME=C:\PROGRA~1\Java\[jdk_version]"
```

4. Once configured, open the files to start the 1Integrate components.

# Deploying the Service

Once you have checked your JAVA_HOME variable, you can deploy 1Integrate as a service by running the following scripts:

> **Note:** Both scripts must be run by an admin user account.

### install-Interface-Service.cmd

The interface script will create a service with the name "**1INTService**" and the display name "**1Integrate Interface [version]**".

---

**install-Engine-Service.cmd**

The engine script will automatically create *N* services with the name
"**1INTEngine[N]Service**" and the display name "**1Integrate Engine [version]
#[N]**" based on how many engines have already been configured.

## Configure additional engines (Service)

1. Create a copy of the **standalone-engine** folder and rename it
   **standalone-engine2** etc.

   > ⊟ **Note:** You must suffix standalone-engine with an integer.

2. Inside the cloned directory, open the **configuration\engine.conf.bat**

3. Update `SET "ENGINE_NUMBER=1"` to the corresponding number e.g
   `SET "ENGINE_NUMBER=2"`

4. You are now ready to start the **install-Engine-Service.cmd** script, which
   will deploy all configured engines.

> ⊟ **Note:** Along with the installation there are two uninstallation scripts.
> You will not need to define `JAVA_HOME`, as per the installation script.
> Uninstallation must be carried out prior to an upgrade, using the scripts
> packaged with the current installation, not the upgrade.

You can label Engines you are deploying to group them, and control which
sessions run on different groups.

Windows:

1. Open the **standalone-engine** directory for the engine you will be
   applying labels to.

2. Within the **configuration** directory, open the **engine.conf.bat** file.

3. Locate and uncomment the following lines:

   `SET "engine.labels="`

4. Add engine labels as a comma separated list:

   `SET "engine.labels=label1,label2,label3"`

5. Repeat the steps for each engine being deployed.

# 5 Testing the Installation

> **Note:** Empty your browser cache before testing your installation.

1Integrate can be accessed through the following site:
http://[machine]:[service_port]/1Integrate

> **Note:** Use the port number specified during installation, by default this is 8080.

Log in as a user with administrator permissions.

To verify the engine installations, click the **Admin** tab and check that the Grid Topology matches the number of interfaces and engines installed.

If you need to access the 1Integrate Web Services API, then a web page with documentation and a link to the WSDL document can be found at:
http://[machine]:[service_port]/soap

# 6 Upgrading an Installation

> **Note:** Before performing an upgrade to an existing installation, ensure **all sessions are stopped** and perform a **backup** of your repository.

> **Note:** These instructions apply when upgrading from one version to the immediately subsequent release only (e.g. from 1.1 to 1.2). If you are performing an upgrade from any older version, please consult your release notes or contact 1Spatial Support.

## Upgrade an Installation:

Upgrading an installation on Wildfly consists of unpacking your new release folder, copying across your existing settings files, and editing the Java location in your interface and engine component files.

> **Note:** The following instructions use an example where currently version 1.A is installed, and we want to upgrade to 1.B. We have used **C:\1Spatial** as a directory within which to store our installation package files, organised into **C:\1Spatial\Product-1.A** and **C:\1Spatial\Product-1.B**.

1. Unzip your new Release Package, and the Wildfly folder within it (e.g. **C:\Program Files\1Spatial\Product-1.B\Product-1.B_wildfly\wildfly-[version]**).

2. Copy the following properties files from the **SETTINGS** folder in your *existing* installation directory (e.g. **C:\Program Files\1Spatial\Product-1.A\Product-1.A_wildfly\wildfly-[version]\SETTINGS**, into the **SETTINGS** folder in your *new* installation directory:

   ▶ **settings.properties**
   ▶ **roles.properties**
   ▶ **users.properties**

3. Copy any built-ins and custom data stores (**.jar** files) from your *existing* installation directory (e.g. **C:\Program Files\1Spatial\Product-1.A\Product-1.A_wildfly\wildfly-[version]\standalone\deployments\Product-1.A.ear\APP-INF\lib** and

**C:\Program Files\1Spatial\Product-1.A\Product-1.A_wildfly\wildfly-[version]\standalone-engine\deployments\Product-1.A.ear\APP-INF\lib**) into the corresponding **lib** folders within your new installation directory.

4. Before running your start-up scripts, you must first ensure your deployment is configured to refer to the correct version of Java. For the Java version supported for your release of 1Integrate, please refer to the release notes.

# JAVA_HOME

Once you are sure you have the correct version of Java installed, you will need to ensure that the JAVA_HOME variable is defined.

> **Note:** If the JAVA_HOME variable is already defined on your system with the correct version of Java, then you do not need to do anything.

If the variable is not set, first check that the default option in the **start-Interface** and **start-Engine** point to the correct location:

**Windows**

```
SET "DEFAULT_JAVA_HOME=C:\PROGRA~1\Java\jdk
[version_number]"
```

**Linux**

```
export DEFAULT_JAVA_HOME="/usr/java/jdk[version_
number]"
```

If the default is the wrong path to your Java, then carry out the following steps:

Windows:

1. Edit **start-Interface.cmd** and **start-Engine.cmd**.

2. Locate the following lines:

```
@REM Uncomment and set this path to be
correct for your environment
REM SET "JAVA_HOME=C:\PROGRA~1\Java\[jdk_
version]"
```

3. Uncomment the second line and remove "REM", so that it begins with "SET". Ensure that it is pointing to the location of Java home for

the correct version of Java installed as part of the pre-requisites. The resulting line should appear as follows:

```
SET "JAVA_HOME=C:\PROGRA~1\Java\[jdk_
version]"
```

4. Once configured, open the files to start the 1Integrate components.

## Linux:

1. Edit **start-Interface.sh** and **start-Engine.sh**.

2. Uncomment the line just below "`#Uncomment and set this path to be correct for your environment`":

```
export JAVA_HOME="/usr/java/[jdk_version]"
```

3. Once configured, open the files to start the 1Integrate components.