IBM webMethods Package Manager CLI Reference

Version 11.1

September 2024

Document ID: IS-WPM-REF-111-20240913

Table of Contents

IBM webMethods Package Manager CLI Reference

The purpose of this guide is to explain how to use IBM webMethods Package Manager and outline the commands you can use to operate it effectively.

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
Narrowfont	Identifies service names and locations in the format *folder.subfolder.service*, APIs, Java classes, methods, properties.
Italic	Identifies: Variables for which you must supply values specific to your own situation or environment. New terms the first time they occur in the text. References to other documentation sources.
`Monospace font`	Identifies: Text you must type in. Messages displayed by the system. Program code.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

1 What Is the IBM webMethods Package Manager Command Line Interface?

IBM webMethods Package Manager (wpm) is a command line interface for sharing Integration Server or Microservices Runtime packages. Administrators and developers can use wpm to download and install packages from a Git repository or a central package registry such as the IBM webMethods Package Registry. The Package Registry is a set of repositories that reference both official IBM licensed modules and public packages for IBM webMethods.

With wpm, you can leverage the pre-built images provided by IBM in the Docker repository at https://containers.webmethods.io/products. Use wpm to quickly add packages to the pre-built images for Integration Server or Microservices Runtime without needing to use IBM webMethods Installer or IBM webMethods Update Manager.

wpm is available from the following:

- The pre-built Docker images for Integration Server or Microservices Runtime provided by IBM at https://containers.webmethods.io/products.
- IBM webMethods Installer

2 wpm Requirements

Using the wpm CLI requires the following:

- wpm CLI must be installed. wpm is at the following location in the pre-built Docker images provided by IBM at https://containers.webmethods.io/products or when installed with Integration Server or Microservices Runtime via the IBM webMethods Installer: *Software AG_directory*/wpm.
- One of the following:
  - A GitHub account for accessing packages in a GitHub repository.
  - An Empower account for accessing the private, "licensed" repository in IBM webMethods Package Registry at https://packages.webmethods.io and an access token for the repository. The "licensed" repository references packages provided by IBM that are officially supported as part of your contract.

    You can create an access token in the IBM webMethods Package Registry web portal and use that token to authenticate your account. Supply the token when executing wpm CLI commands or add it to the wpm.yml configuration file.

    For more information about creating an access token for the "licensed" repository, see the IBM webMethods Package Registry help.
- The operating system environment variable JAVA_HOME must point to a valid JVM directory. wpm requires Java 11 or higher.

3 wpm Configuration

During command execution, wpm obtains configuration values from settings specified in the following locations:

- You can set values in command argument using the command line
- wpm.yml configuration file

Values specified in an argument at the command line override the value set for a corresponding property in the wpm.yml configuration file.

The wpm.yml Configuration File

The wpm.yml file is the built-in configuration file for wpm. You can provide configuration information in the wpm.yml file to avoid providing it in an argument for a wpm command. For example, you can specify the `target_installation` location in the wpm.yml file instead of providing the `-d` argument for a command. You can also specify the package repositories from which wpm can install or update packages instead of providing the repository location and authentication information per command. Providing configuration in the wpm.yml file decreases the information that you need to provide at the command line, simplifying the wpm CLI.

The wpm.yml file included with wpm appears below and is located at: *Software AG_directory*/wpm.

```
version: 1
# Command line argument that overrides each key is given in parentheses.
switches:
  # Relative path of the target installation of IS. (-d)
  target_installation:
  # Scan branches for versions. (-sb)
  scan_branches: true
  # Delete contents related to source code management from target directory on a
successful install or update. (-kr)
  cleanup: true
# Define your remote registries here (one or multiple, wpm or git)
# and reference them using -r from the command line interface.
repositories:
  default:
    type: wpr
    location: https://packages.webmethods.io
    working_dir: .
```

By default, the wpm.yml file includes a repository declaration for the IBM webMethods Package Registry, which is the official package registry hosted by IBM. The registry contains IBM packages that are available for installation. You can add other repositories declarations such as a GitHub repository or a private registry. For more information about declaring repositories, see "Declaring Remote Repositories" on page 13.

For more information about using IBM webMethods Package Registry, see the IBM webMethods Package Registry help at https://docs.webmethods.io/wpm/getting-started/home/index.html.

To use a customized wpm.yml file with an Integration Server or Microservices Runtime in a Docker container, you can add the wpm.yml file to the base image (Docker image). The Docker images provided by by IBM in the Docker repository at https://containers.webmethods.io/products include the default wpm.yml configuration file.

Declaring Remote Repositories

Declare the remote repositories and/or package registries that you want to use with wpm in the wpm.yml configuration file. For each remote repository you can declare the name, type, location, and authentication credentials.

Follow this structure when declaring a remote repository:

```
respositories:
   name:
    type:
    location:
    registry:
    creds:
      user:
      password:
      token:
    working_dir:
```

Where:

- `name` - Name for the repository that can be referenced from the `-r` argument. The name used in the wpm.yml file does not need to match the name of the actual repository or registry.
- `type` - Type of repository. Possible values are:
  - `wpr` - IBM webMethods Package Registry
  - `git`
- `location` - URL to Git repository or package registry
- `creds` - Credentials for accessing Git repository or package registry. Specify one or more of the following:
  - `user`
  - `password`
  - `token`
  - `private_key_path`
  - `passphrase`
- `working_dir` - Relative or absolute path to the directory to use for activities, such as interim repository cloning and branch check outs. As part of completing a command, wpm moves content from the working directory to the destination folder in the `target_installation`. The relative path is relative to the location of the wpm launch script (usually \wpm\bin).

4 Running wpm

Before running wpm, make sure you have set the JAVA_HOME operating system environment variable.

wpm is at the following location when installed with Integration Server or Microservices Runtime via the IBM webMethods Installer: *Software AG_directory*/wpm.

In the pre-built Docker images of Integration Server or Microservices Runtime provided by IBM at https://containers.webmethods.io/products, you can run wpm from anywhere by simply typing `wpm`.

To start wpm in an on-premise installation

1. Open a command prompt and navigate to the wpm/bin directory.
2. Execute wpm.bat/wpm.sh.

5 wpm CLI Commands

The wpm CLI uses a command line structure that you specify in the following order:

1. Base call to wpm
2. Top-level command
3. CLI options required or used by the operation

While this guide describes wpm CLI commands, wpm includes command line help. Type `wpm` at the command prompt to display the command line help.

wpm install

Downloads and installs specified Integration Server packages from a source repository into a target installation. Packages can either be the entire contents of the repository or subfolders in the repository.

Synopsis

```
wpm install [package-name:version-number package-name2:version-number …
package-nameN:version-number] [optional arguments]
```

where *package-name* is a space-separated list of Integration Server packages to install.

Specifying *version-number* is optional. If *version-number* is not specified, IBM webMethods Package Manager looks for the highest numerical version number available in the source repository. IBM webMethods Package Manager first looks in the tags and branches, looking in branches only if `-sb=true` (the default), and then falls back to the main branch.

Arguments

The following table lists all available arguments for the `wpm install` command and their descriptions:

Argument	Description
`-r [repository]`	The repository from which to install packages. The repository can be an absolute path to a GitHub repository, a URL, or an alias of a repository defined in the wpm.yml configuration file. Optional if a single repository is defined in the wpm.yml configuration file. Required if the wpm.yml file does not define any repositories or defines more than one repository.
`-u [username]`	Username for the repository from which to install packages. For a GitHub repository, specify a GitHub user name. For a Package Registry repository, specify an Empower username. Optional. Required when installing packages from a private GitHub repository or a private repository in the Package Registry.

Argument	Description
`-p [password]`	When `creds\private_key_path` is set in the wpm.yml file or in the `-k` argument, specify a passphrase in wpm.yml configuration file or in the `-p` argument.
`-k [private-key]`	Relative or absolute path to the location of the private key file to use for SSH authentication with the remote GitHub repository. The relative path is relative to the location of the wpm launch script (usually \wpm\bin). Required if SSH authentication is used and the `creds\private_key_path` property is not set in the wpm.yml configuration file.
`-j [token]`	OAuth token for obtaining packages from the GitHub repository or IBM webMethods Package Registry protected by token authorization. Required only if remote GitHub repository or IBM webMethods Package Registry repository requires an OAuth token for authentication and `jwt_token` is not set in the wpm.yml file.
`-ad {true|false}`	Whether IBM webMethods Package Manager checks for dependent packages and automatically installs them. If set to `false` or not present, wpm does not attempt to automatically install dependencies. All missing dependencies are listed as warnings in the output. If set to `true`, wpm attempts to install all dependent packages and reports errors for any it cannot install. Optional. The default is `false`.
`-sb {true|false}`	Whether IBM webMethods Package Manager scans branches and tags in the repository for the specified version number of a packages. If set to `true`, wpm scans branches and tags; `false` scans only tags. Optional. The default is `true`.
`-kr {true|false}`	Whether IBM webMethods Package Manager keeps the local repository or cleans it up after the `install` command completes. A value of `true` keeps the repository; `false` indicates that webMethods Package Manager deletes all the local repositories once the `install` command completes successfully. Keeping the local repository can be useful if multiple instances of Integration Server reside on the same machine and you want to update the packages in multiple instances or if different versions of the same package will be installed. Use `wpm clean` to delete local repositories. Optional. The default is `false`.
`-ws [package-manager-server]`	URL of the IBM webMethods Package Registry that contains the package to install.

Argument	Description
	Optional. Specify `-ws` argument when you want to install a package located in the IBM webMethods Package Registry and one of the following is true: ■ The wpm.yml configuration file does not set a IBM webMethods Package Registry. ■ You want to override the value configured in the wpm.yml configuration file. ■ A wpm.yml file is not used with wpm.
`-wr [package-manager-registry]`	Name of the IBM webMethods Package Registry from which to install packages. The IBM webMethods Package Registry can contain multiple registries in addition to the default registry. Optional. Supply `-wr` if also supplying `-ws` and you want to install packages from a registry other than the default registry. If the `-wr` argument is not supplied but the `-ws` argument is supplied, wpm uses the default registry for the install operation.
`-d [target-installation]`	Relative or absolute path to the target server installation. The relative path is relative to the location of the wpm launch script (usually \wpm\bin). Required if all of the following are true: ■ The `target_installation` property is not set in the wpm.yml file. ■ The wpm.home environment variable is not specified. ■ IBM webMethods Package Manager is not in the same installation directory as the target server.
`-ks {true|false}`	Whether IBM webMethods Package Manager preserves the enabled or disabled state of a package. If the `-ks` argument is not supplied, wpm sets the state of a package as enabled. The `-ks` argument also affects any dependent packages. Optional. The default is `false`.

Usage Notes

- Argument values specified in the command override related property values in the wpm.yml configuration file.
- The `wpm install` command does not activate the installed packages. You must restart Integration Server to activate the package.

- The `wpm install` command checks for any dependent packages and, if possible, installs them based on the value of the `-ad` argument. If wpm cannot install the dependent packages, it returns an error or a warning based on the value of the `-ad` argument.
- The `wpm install` command places the installed packages in *Integration Server_directory*/packages. It does not place packages in *Integration Server_directory*/*`instance_name`*/replicate/inbound.
- During package installation, wpm compares the runtime version of the target Integration Server with the value of the `target_server_version` attribute in the package manifest. (The runtime version of an Integration Server is located in the MANIFEST.MF file located in *Integration Server_directory*\lib\wm-isserver.jar). wpm does this to ensure that the functionality in the package is available in the target Integration Server. One of the following occurs during comparison:
  - If the `target_server_version` attribute is present in the package and the destination Integration Server's runtime version is higher than or equal to the attribute value, wpm installs the package into the destination folder.
  - If the `target_server_version` attribute is present in the package and the destination Integration Server's runtime version is lower than the attribute value, wpm cancels the install operation and adds an error to the operation result.
  - If the `target_server_version` attribute is present in the package but wpm cannot determine the runtime version of the destination Integration Server, wpm cancels the install operation and adds an error to the operation result. wpm might be unable to detect a version number due to reasons such as an invalid destination location, a missing JAR file, or a corrupted JAR file.
  - If the `target_server_version` attribute is not present in the package, wpm does not check the runtime version of the destination Integration Server. Instead, wpm installs the package and writes a warning message in the operation result. The `target_server_version` attribute might be absent if the package was created in an Integration Server version that does not support the `target_server_version` attribute, which was added in Integration Server version 11.1.

Examples Which Do Not Use the wpm.yml Configuration File

The following examples of the `wpm install` command outline common use cases which do not use the wpm.yml configuration file:

- To install version 1.0.0 of package called PackageA from the specified GitHub repository into the specified target installation:

```
wpm install PackageA:1.0.0 -r https://github.com/[username]/ -u [username] -p
[personal-access-token] -d [target-installation]
```

  wpm accesses the GitHub repository using the provided username and token. When cloning from GitHub, wpm clones the https://github.com/[username]/PackageA repository.

  If the repository contains a manifest.v3 file at the top level, wpm downloads the entire repository contents as the package body. Alternatively, if the repository does not contain the manifest.v3 file at the top level, wpm looks for a folder named PackageA within the repository.

- To install the highest version of PackageA from the specified GitHub repository into the specified target installation:

  ```
  wpm install PackageA -r https://github.com/[username]/ -u [username] -p
  [personal-access-token] -d [target-installation]
  ```

  wpm accesses the GitHub repository using the provided username and personal access token. When cloning from GitHub, wpm clones the https://github.com/[*username*]/PackageA repository.

  When resolving which version of the PackageA to install, wpm first looks for the highest numerical version in the repository tags and branches. If wpm cannot locate a version of PackageA in the tags and branches, wpm falls back to the main branch of the repository.

- To install PackageA version 1.0.0 and the highest versions of PackageB and PackageC from the specified GitHub repository into the specified target installation:

  ```
  wpm install PackageA:1.0.0 PackageB PackageC -r https://github.com/[username]/ -u
   [username] -p [git-token] -d [target-installation]
  ```

  wpm accesses the GitHub repository using the supplied username and token `[git-token]`.

- To install the latest version of PackageA from the specified GitHub repository into the specified target installation:

  ```
  wpm install PackageA -r https://github.com/[username]/ -k [private-key-location]
  -p [passphrase] -d [target-installation]
  ```

  wpm accesses the GitHub repository using the private key file located at `[private-key-location]` and passphrase specified in `[passphrase]`.

- To install PackageA from the GitHub repository `[git-repo]` into the specified target installation:

  ```
  wpm install PackageA -r [git-repo] -j [oauth-token] -d [target-installation]
  ```

  wpm uses the provided OAuth token `[oauth-token]` to access the repository.

- To install the WmJDBCAdapter package from the default IBM webMethods Package Registry repository into the specified target installation:

  ```
  wpm install WmJDBCAdapter -ws https://packages.webmethods.io -wr licensed -u
  [empower-user] -p [empower-password] -d [target-installation]
  ```

  wpm uses the Empower credentials `[empower-user]` and `[empower-password]` to log into the registry. wpm resolves the GitHub location information using the information contained in the registry and then proceeds to download and install the package.

- To install the WmJDBCAdapter package from the default IBM webMethods Package Registry repository into the target installation:

  ```
  wpm install WmJDBCAdapter -ws https://packages.webmethods.io -u [empower-user] -p
   [empower-password] -d [target-installation]
  ```

  wpm uses the supplied Empower credentials `[empower-user]` and `[empower-password]` to access the registry.

- To install the WmJDBCAdapter package from the IBM webMethods Package Registry into the specified target installation:

```
wpm install WmJDBCAdapter -ws https://packages.webmethods.io -wr licensed -j
[oauth-token] -d [target-installation]
```

  wpm uses the supplied OAuth token to access the registry.

Examples Which Use the wpm.yml Configuration File

The following examples of the `wpm install` command rely on information from the wpm.yml configuration file:

- To install package PackageA from the GitHub repository my_packages for which connection information, including location, is defined in the wpm.yml configuration file:

```
wpm install PackageA -r my_packages
```

  Example wpm.yml configuration file:

```
  my_packages:
    type: git
    location: https://github.com/[username]
    creds:
      user: [username]
      password: [personal-access-token]
    working_dir: [working-dir-location]
```

- To install package PackageA from a repository configured in the wpm.yml file into an target installation configured in the wpm.yml file:

```
wpm install PackageA
```

  This approach works if the wpm.yml configuration file contains only one defined repository.

- To install the WmJDBCAdapter package from the registry defined in the repositories section of the wpm.yml configuration file into the installation location specified in the wpm.yml configuration file:

```
wpm install WmJDBCAdapter
```

  wpm uses the connection information in the `repositories\wpm` properties to connect to and access the registry.

  Example wpm.yml configuration file:

```
repositories:
  wpm:
    type: wpr
    location: https://packages.webmethods.io
    registry: licensed
    creds :
      token : [oauth-token]
    working_dir: ./Temp/repositories/
```

wpm update

Updates the listed packages in the target installation to the specified version. If you do not specify a version, wpm updates the package to the highest version available in the source repository.

Synopsis

```
wpm update [package-name:version-number package-name2:version-number …
package-nameN:version-number] [optional arguments]
```

where *package-name* is a space-separated list of Integration Server packages to update.

The *version-number* is optional. If *version-number* is not specified, IBM webMethods Package Manager looks for the highest numerical version number available in the source repository. IBM webMethods Package Manager first looks in the tags and branches, looking in branches only if `-sb=true` (the default), and then falls back to the main branch.

Arguments

The following table lists all available arguments available for the `wpm update` command and their descriptions:

Argument	Description
`-r [git_repo]`	The repository from which to update packages. The repository can be an absolute path to a GitHub repository, a URL, or an alias of a repository defined in the wpm.yml configuration file. Optional if a single repository is defined in the wpm.yml configuration file. Required if the wpm.yml file does not define any repositories or defines more than one repository.
`-u [username]`	Username for the source repository from which to install packages. For a GitHub repository, specify a GitHub user name. For an IBM webMethods Package Registry repository, specify an Empower username. Optional. Required when installing packages from a private GitHub repository or private IBM webMethods Package Registry repository.
`-p [password]`	Password, passphrase, or token for connecting to the source repository. Required if password is not defined in the `creds\password` property or `creds\passphrase` property in the wpm.yml configuration file. When working with a GitHub repository using basic authentication, specify a GitHub developer token. When working with a GitHub repository using private keys, specify a passphrase.

Argument	Description
	When working with IBM webMethods Package Registry, specify an Empower password. When `creds\private_key_path` is set in wpm.yml file or in -k argument, specify passphrase in wpm.yml configuration file or in the `-p` argument.
`-k [private_key]`	Relative or absolute path to the location of the private key file to use for SSH authentication with the remote GitHub repository. The relative path is relative to the location of the wpm launch script (usually \wpm\bin). Required if SSH authentication is used and the `creds\private_key_path` property is not set in the wpm.yml configuration file.
`-j [token]`	OAuth token for obtaining packages from the GitHub repository or IBM webMethods Package Registry repository protected by token authorization. Required only if remote GitHub repository or registry requires an OAuth token for authentication and `jwt_token` is not set in the wpm.yml file.
`-ad {true|false}`	Whether IBM webMethods Package Manager checks for dependent packages and automatically updates them. If set to `false` or not present, wpm does not attempt to automatically update dependencies. All missing dependencies are listed as warnings in the output. If set to `true`, wpm attempts to update all dependent packages and reports errors for any it cannot update. Optional. The default is `false`.
`-sb {true|false}`	Whether IBM webMethods Package Manager scans branches and tags in the repository for the specified version number of a packages. If set to `true`, wpm scans branches and tags; `false` scans only tags. Optional. The default is `true`.
`-kr {true|false}`	Whether IBM webMethods Package Manager keeps the local repository or cleans it up after the `wpm update` command completes. A value of `true` keeps the repository; `false` indicates that IBM webMethods Package Manager deletes all the local repositories once the `wpm update` command completes successfully. Keeping the local repository can be useful if multiple instances of Integration Server reside on the same machine and you want to update the packages in multiple instances or if different versions of the same package will be installed. Use `wpm clean` to delete local repositories. Optional. The default is `false`.

Argument	Description
`-ws [package-manager-server]`	URL of the IBM webMethods Package Registry that contains the package to update. Optional. Specify `-ws` argument when you want to update a package located on a IBM webMethods Package Registry and one of the following is true: ■ The wpm.yml configuration file does not set a IBM webMethods Package Registry. ■ You want to override the value configured in the wpm.yml configuration file. ■ A wpm.yml file is not used with wpm
`-wr [package-manager-registry]`	Name of the registry in IBM webMethods Package Registry from which to update packages. The IBM webMethods Package Registry can contain multiple registries in addition to the default registry. Optional. Supply `-wr` if also supplying `-ws` and you want to update packages from a registry other than the default registry. If the `-wr` argument is not supplied but the `-ws` argument is supplied, wpm uses the default registry for the update operation.
`-d [target-installation]`	Relative or absolute path to the target installation server. The relative path is relative to the location of the wpm launch script (usually \wpm\bin). Required if all the following are true: ■ The `target_installation` property is not set in the wpm.yml configuration file. ■ The SAG home environment variable is not specified. ■ IBM webMethods Package Manager is not in the same installation directory as the target server.
`-ks {true|false}`	Whether IBM webMethods Package Manager preserves the enabled or disabled state of a package. If the `-ks` argument is not supplied, wpm sets the state of a package as enabled. The `-ks` argument also affects any dependent packages. Optional. The default is `false`.

Usage Notes

- The `wpm update` command does not activate the updated packages. You must restart Integration Server to activate the package.

- The `wpm update` command checks for any dependent packages and, if possible, updates them based on the value of the `-ad` argument. If wpm cannot update the dependent packages, it returns an error or a warning based on the value of the `-ad` argument.
- The `wpm update` command places the updated packages in *Integration Server_directory*/packages. It does not place packages in *Integration Server_directory*/`[instance_name]`/replicate/inbound.
- During package update, wpm compares the runtime version of the target Integration Server with the value of the `target_server_version` attribute in the package manifest. (The runtime version of an Integration Server is located in the MANIFEST.MF file located in *Integration Server_directory*\lib\wm-isserver.jar.) wpm does this to ensure that the functionality in the package is available in the target Integration Server. One of the following occurs during comparison:
  - If the `target_server_version` attribute is present in the package and the destination Integration Server's runtime version is higher than or equal to the attribute value, wpm updates the package in the destination folder.
  - If the `target_server_version` attribute is present in the package and the destination Integration Server's runtime version is lower than the attribute value, wpm cancels the update operation and adds an error to the operation result.
  - If the `target_server_version` attribute is present in the package but wpm cannot determine the runtime version of the destination Integration Server, wpm cancels the update operation and adds an error to the operation result. wpm might be unable to detect a version number due to reasons such as an invalid destination location, a missing JAR file, or a corrupted JAR file
  - If the `target_server_version` attribute is not present in the package, wpm does not check the runtime version of the destination Integration Server. Instead, wpm updates the package and writes a warning message in the operation result. The `target_server_version` attribute might be absent if the package was created in an Integration Server version that does not support the `target_server_version` attribute, which was added in Integration Server version 11.1.
- Argument values specified in the command override related property values in the wpm.yml configuration file.

wpm clean

Cleans up the specified local repository or all repositories in the working directory.

Synopsis

```
wpm clean [optional arguments]
```

Arguments

The following table lists all available arguments for the `wpm clean` command and their descriptions:

Argument	Description
`-r [repository]`	Optional. Comma-separated list of local repositories to clean up. If `-r` is not specified, wpm cleans up all local repositories specified in the wpm.yml configuration file. If the wpm.yml file does not specify any local repositories, wpm attempts to clean up only the local repository.

Usage Notes

- When `wpm install` or `wpm update` fails, clean up of local repositories is not done automatically. Use the `wpm clean` command.
- wpm also keeps local repositories when `-kr=true` for `wpm install` or `wpm update`.
- Argument values specified in the command override related property values in the wpm.yml configuration file.

wpm remove

Removes specified packages from the target installation.

Synopsis

```
wpm remove [package-name package-name2 … package-nameN] [optional arguments]
```

where *package-name* is a space-separated list of Integration Server packages to remove.

Arguments

The following table lists all available arguments available for the `wpm remove` command and their descriptions:

Argument	Description
`-d [target-installation]`	Relative or absolute path to the target server installation from which to remove packages. The relative path is relative to the current working directory. Required if all of the following are true: - The `target_installation` property is not set in the wpm.yml coniguration file. - The SAG home environment variable is not specified. - IBM webMethods Package Manager is not in the same installation directory as the target server.

Usage Note

Argument values specified in the command override related property values in the wpm.yml configuration file.