JavaServiceWrapper

Windows platform integration of Tanuki Software's Java Wrapper Service facility

Table of contents

1 Overview....3
2 Design....3
3 Constraints....3
3.1 Allowed Child Dependencies....3
3.2 Allowed Parent Dependencies....4
4 Attributes....4
4.1 Exported Attributes....4
4.2 Defaults for Imported Attributes....4
5 Commands....5
5.1 Deploy....5
5.2 Add....5
5.3 Remove....6
5.4 Service-Install....6
5.5 dispatchServiceInstall....7
5.6 dispatchServiceConfigure....7
5.7 dispatchServicePackagesInstall....7
5.8 Docs-Generate....7
5.9 addService....8
5.10 removeService....8
5.11 assertServiceIsUp....8
5.12 assertServiceIsDown....9
5.13 assertServiceIsInstalled....9

5.14 assertServiceIsNotInstalled....10
5.15 startService....10
5.16 stopService....11
6 Related Types....11
6.1 JavaServiceWrapperSetting....11
6.2 JavaServiceWrapperJavaHome....11
6.3 JavaServiceWrapperJavaMainClass....12
6.4 JavaServiceWrapperJavaClassPath....13
6.5 JavaServiceWrapperJavaAdditional....14
6.6 JavaServiceWrapperJavaInitMemory....15
6.7 JavaServiceWrapperJavaMaxMemory....15
6.8 JavaServiceWrapperAppParameters....16
6.9 JavaServiceWrapperConsoleTitle....17
6.10 JavaServiceWrapperNtServiceName....18
6.11 JavaServiceWrapperNtServiceDisplayName....19
6.12 JavaServiceWrapperNtServiceDescription....19
6.13 JavaServiceWrapperNtServiceStartType....20
6.14 JavaServiceWrapperNtServiceInteractive....21
6.15 JavaServiceWrapperNtServiceAccount....22
6.16 JavaServiceWrapperNtServicePassword....23

1. Overview

JavaServiceWrapper: *Windows platform integration of Tanuki Software's Java Wrapper Service facility*

This type manages the Tanuki Software Java Wrapper Service, and allows a dependent Service object to be automatically deployed.

A JavaServiceWrapperZip object should be added as a child dependency.

A Service object should also be added as a child dependency.

Appropriate Settings should be added to JavaServiceWrapper object to configure the appropriate Java based configuration to run the underlying Service.

An example of using JavaServiceWrapper for managing the JBossServer type is here: Windows service example.

See: Tanuki Software Ltd. - Java Service Wrapper.

2. Design

Super Type
Service

Role	**Concrete**. (Objects can be created.)
Instance Names	**Unique**
Notification	**false**
Template Directory	
Data View	Children, proximity: **1**
Logger Name	

3. Constraints

3.1. Allowed Child Dependencies

- JavaServiceWrapperAppParameters 1
- JavaServiceWrapperConsoleTitle 1
- JavaServiceWrapperJavaAdditional 1
- JavaServiceWrapperJavaClassPath 1

- JavaServiceWrapperJavaHome 1
- JavaServiceWrapperJavaInitMemory 1
- JavaServiceWrapperJavaMainClass 1
- JavaServiceWrapperJavaMaxMemory 1
- JavaServiceWrapperNtServiceDescription 1
- JavaServiceWrapperNtServiceDisplayName 1
- JavaServiceWrapperNtServiceInteractive 1
- JavaServiceWrapperNtServiceName 1
- JavaServiceWrapperNtServiceStartType 1
- JavaServiceWrapperZip 1
- Service1

1: These types have a *Singleton* constraint. Only one instance may be added as a resource.

3.2. Allowed Parent Dependencies

- Node
- Site

4. Attributes

4.1. Exported Attributes

Name	Property	Description
`installRoot`	deployment-install-root	The deployment-install-root is exported as the "installRoot" attribute. It is used to default the -installroot option in many commands.

4.2. Defaults for Imported Attributes

Name	Default
`javaServiceWrapperConsoleTitle`	Test Wrapper Sample Application
`javaServiceWrapperJavaHome`	${env.JAVA_HOME}
`javaServiceWrapperJavaInitMemory`	3
`javaServiceWrapperJavaMainClass`	org.tanukisoftware.wrapper.WrapperSimpleApp
`javaServiceWrapperJavaMaxMemory`	64

javaServiceWrapperNtServiceDescriptic	Test Wrapper Sample Application Description
javaServiceWrapperNtServiceDisplayNam	Test Wrapper Sample Application
javaServiceWrapperNtServiceInteractiv	false
javaServiceWrapperNtServiceName	testwrapper
javaServiceWrapperNtServiceStartType	AUTO_START

5. Commands

Note:
Commandline options displayed in square brackets "[]" are optional. If an option expects arguments, then angle brackets are shown after the option "<>". Any default value is shown within the brackets.

5.1. Deploy

Run the service deployment cycle, stopping, unconfiguring, installing package dependencies, installing the dependent Service, and configuring and the starting the service. .

This implementation overrides the default Deploy command to insert the Service-Install command after the Packages-Install command. This allows the JavaServiceWrapper object to perform selective parts of the normal Deploy workflow for its dependent Service child, without invoking the normal Start and Stop lifecycle commands for the Service.

The JavaServiceWrapper's Start and Stop commands invoke the Tanuki Software Java Service Wrapper start/stop commands, which invoke the Windows Service management operations to directly start or stop the underlying Java based Service software.

Usage
```
Deploy
```

5.1.1. Workflow

1. Stop
2. Remove
3. Packages-Install
4. Service-Install
5. Configure
6. Add
7. Start

5.2. Add

Add the Java Service Wrapper Windows service if it is not already installed.

This workflow idempotently registers this object as a Windows service using the Tanuki Java Service Wrapper software.

Usage
`Add`

5.2.1. Workflow

1. assertServiceIsInstalled

5.2.2. Error Handler

Command	addService

5.3. Remove

Remove the Java Service Wrapper Windows service if it is installed.

This workflow idempotently de-registers this object as a Windows service using the Tanuki Java Service Wrapper software.

Usage
`Remove`

5.3.1. Workflow

1. assertServiceIsNotInstalled

5.3.2. Error Handler

Command	removeService

5.4. Service-Install

Install a dependent Service object without starting it directly

This workflow command dispatches the necessary commands to the dependent Service object to install it: Install, Packages-Install, and Configure.

Usage
`Service-Install`

5.4.1. Workflow

1. dispatchServiceInstall
2. dispatchServicePackagesInstall
3. dispatchServiceConfigure

5.5. dispatchServiceInstall

Dispatch the Install command to a dependent service object

Dispatch the "Install" command to a dependent Service.

Usage

```
dispatchServiceInstall
```

5.6. dispatchServiceConfigure

Dispatch the Configure command to a dependent service object

Dispatch the "Configure" command to a dependent Service.

Usage

```
dispatchServiceConfigure
```

5.7. dispatchServicePackagesInstall

Dispatch the Packages-Install command to a dependent service object.

Dispatch the "Packages-Install" command to a dependent Service.

Usage

```
dispatchServicePackagesInstall
```

5.8. Docs-Generate

Generate the server properties file

Generates the wrapper.conf file based on the Settings and attribute defaults of this object.

Usage

```
Docs-Generate [-installroot <>] [-packagebase <>]
```

Options

Option	Description
`installroot`	*Java wrapper service wrapper home directory*
`packagebase`	*Java wrapper service wrapper package installation directory*

5.9. addService

Add the Java Service Wrapper Windows service to the system

Directly registers the Windows service using the Java Service Wrapper software.

Usage

```
addService [-installroot <>] [-packagebase <>]
```

Options

Option	Description
`installroot`	*Java wrapper service wrapper home directory*
`packagebase`	*Java wrapper service wrapper package installation directory*

5.10. removeService

Remove the Java Service Wrapper Windows service from the system

Directly de-registers the Windows service using the Java Service Wrapper software.

Usage

```
removeService [-installroot <>] [-packagebase <>]
```

Options

Option	Description
`installroot`	*Java wrapper service wrapper home directory*
`packagebase`	*Java wrapper service wrapper package installation directory*

5.11. assertServiceIsUp

Check whether the Java Service Wrapper Windows service is up

Queries the state of the Windows service using the Java Service Wrapper software, and fails if the state is not "running".

Usage

```
assertServiceIsUp [-installroot <>] [-packagebase <>]
[-servicename <>]
```

Options

Option	Description
`installroot`	*Java wrapper service wrapper home directory*
`packagebase`	*Java wrapper service wrapper package installation directory*
`servicename`	*Java wrapper service wrapper Windows service name*

5.12. assertServiceIsDown

Check whether the Java Service Wrapper Windows service is down

Queries the state of the Windows service using the Java Service Wrapper software, and fails if the state is "running".

Usage

```
assertServiceIsDown [-installroot <>] [-packagebase <>]
[-servicename <>]
```

Options

Option	Description
`installroot`	*Java wrapper service wrapper home directory*
`packagebase`	*Java wrapper service wrapper package installation directory*
`servicename`	*Java wrapper service wrapper Windows service name*

5.13. assertServiceIsInstalled

Check whether the Java Service Wrapper Windows service is installed

Queries the state of the Windows service using the Java Service Wrapper software, and fails if the state is not "installed".

Usage

```
assertServiceIsInstalled [-installroot <>] [-packagebase <>]
[-servicename <>]
```

Options

Option	Description
`installroot`	*Java wrapper service wrapper home directory*

packagebase	*Java wrapper service wrapper package installation directory*
servicename	*Java wrapper service wrapper Windows service name*

5.14. assertServiceIsNotInstalled

Check whether the Java Service Wrapper Windows service is not installed

Queries the state of the Windows service using the Java Service Wrapper software, and fails if the state is "installed".

Usage

```
assertServiceIsNotInstalled [-installroot <>] [-packagebase
<>] [-servicename <>]
```

Options

Option	Description
installroot	*Java wrapper service wrapper home directory*
packagebase	*Java wrapper service wrapper package installation directory*
servicename	*Java wrapper service wrapper Windows service name*

5.15. startService

Start the Java Service Wrapper Windows service

Directly starts the Windows service using the Java Service Wrapper software.

Usage

```
startService [-installroot <>] [-packagebase <>]
[-servicename <>]
```

Options

Option	Description
installroot	*Java wrapper service wrapper home directory*
packagebase	*Java wrapper service wrapper package installation directory*
servicename	*Java wrapper service wrapper Windows service*

	name

5.16. stopService

Stop the Java Service Wrapper Windows service

Directly stops the Windows service using the Java Service Wrapper software.

Usage

```
stopService [-installroot <>] [-packagebase <>]
[-servicename <>]
```

Options

Option	Description
`installroot`	*Java wrapper service wrapper home directory*
`packagebase`	*Java wrapper service wrapper package installation directory*
`servicename`	*Java wrapper service wrapper Windows service name*

6. Related Types

The following types are defined for use with JavaServiceWrapper.

6.1. JavaServiceWrapperSetting

6.1.1. Overview

JavaServiceWrapperSetting: *Java Service Wrapper configuration setting*

This abstract Setting subtype is the supertype for all JavaServiceWrapper Setting types.

6.1.2. Design

Super Type

Setting

Role	**Abstract**. (Objects cannot be created.)
Instance Names	**Unique**

6.2. JavaServiceWrapperJavaHome

6.2.1. Overview

JavaServiceWrapperJavaHome: *Value of JAVA_HOME for the Java Service Wrapper*

6.2.2. Design

Super Type
JavaServiceWrapperSetting

Role	**Concrete**. (Objects can be created.)
Instance Names	**Unique**

6.2.3. Constraints

6.2.3.1. Allowed Parent Dependencies

- JavaServiceWrapper

6.2.4. Attributes

6.2.4.1. Exported Attributes

Name	Property
`javaServiceWrapperJavaHome`	settingValue

6.3. JavaServiceWrapperJavaMainClass

6.3.1. Overview

JavaServiceWrapperJavaMainClass: *Class implementing the Java Service Wrapper WrapperListener interface*

The main class to set with this Setting is most often one provided by the Tanuki software, such as "org.tanukisoftware.wrapper.WrapperSimpleApp", unless target Java software to manage has its own implementation.

See: Java Service Wrapper - Integration Methods.

6.3.2. Design

Super Type
JavaServiceWrapperSetting

Role	**Concrete**. (Objects can be created.)
Instance Names	**Unique**

6.3.3. Constraints

6.3.3.1. Allowed Parent Dependencies

- JavaServiceWrapper

6.3.4. Attributes

6.3.4.1. Exported Attributes

Name	Property
`javaServiceWrapperJavaMainClass`	settingValue

6.4. JavaServiceWrapperJavaClassPath

6.4.1. Overview

JavaServiceWrapperJavaClassPath: *A list of additional CLASSPATH elements for the Java Service Wrapper, comma separated*

The comma-separated java class paths for this Setting will be added to a default set which includes the necessary jars for the Tanuki service wrapper software.

6.4.2. Design

Super Type
JavaServiceWrapperSetting

Role	**Concrete**. (Objects can be created.)
Instance Names	**Unique**

6.4.3. Constraints

6.4.3.1. Allowed Parent Dependencies

- JavaServiceWrapper

6.4.4. Attributes

6.4.4.1. Exported Attributes

Name	Property
`javaServiceWrapperJavaClassPath`	settingValue

6.5. JavaServiceWrapperJavaAdditional

6.5.1. Overview

JavaServiceWrapperJavaAdditional: *A list of additional JAVA_OPTS for the Java Service Wrapper, space separated*

The values are split at the space characters and generated into the wrapper.conf file as separate numbered properties.

See: wrapper.java.additional<n> Property.

6.5.2. Design

Super Type
JavaServiceWrapperSetting

Role	**Concrete**. (Objects can be created.)
Instance Names	**Unique**

6.5.3. Constraints

6.5.3.1. Allowed Parent Dependencies

- JavaServiceWrapper

6.5.4. Attributes

6.5.4.1. Exported Attributes

Name	Property
`javaServiceWrapperJavaAdditional`	settingValue

6.6. JavaServiceWrapperJavaInitMemory

6.6.1. Overview

JavaServiceWrapperJavaInitMemory: *Initial Java heap size in MB for the Java Service Wrapper*

Used in the wrapper.conf configuration file.

See: wrapper.java.initmemory Property.

6.6.2. Design

Super Type
JavaServiceWrapperSetting

Role	**Concrete**. (Objects can be created.)
Instance Names	**Unique**

6.6.3. Constraints

6.6.3.1. Allowed Parent Dependencies

- JavaServiceWrapper

6.6.4. Attributes

6.6.4.1. Exported Attributes

Name	Property
`javaServiceWrapperJavaInitMemory`	settingValue

6.7. JavaServiceWrapperJavaMaxMemory

6.7.1. Overview

JavaServiceWrapperJavaMaxMemory: *Maximum Java heap size in MB for the Java Service Wrapper*

Used in the wrapper.conf configuration file.

See: wrapper.java.maxmemory Property.

6.7.2. Design

Super Type
JavaServiceWrapperSetting

Role	**Concrete**. (Objects can be created.)
Instance Names	**Unique**

6.7.3. Constraints

6.7.3.1. Allowed Parent Dependencies

- JavaServiceWrapper

6.7.4. Attributes

6.7.4.1. Exported Attributes

Name	Property
`javaServiceWrapperJavaMaxMemory`	settingValue

6.8. JavaServiceWrapperAppParameters

6.8.1. Overview

JavaServiceWrapperAppParameters: *Application commandline parameters for the Java Service Wrapper, space separated.*

The value should be a space separated list of parameters to pass to the main class defined with a JavaServiceWrapperJavaMainClass Setting object.

When the JavaServiceWrapperMainClass value is "org.tanukisoftware.wrapper.WrapperSimpleApp", then the value of this setting should be the Main class name of the target Java application, followed by the commandline parameters separated by spaces.

For example:

```
org.myapp.Main -param value
```

See: Java Service Wrapper - Integration Methods.

See: wrapper.java.mainclass Property.

6.8.2. Design

Super Type
JavaServiceWrapperSetting

Role	**Concrete**. (Objects can be created.)
Instance Names	**Unique**

6.8.3. Constraints

6.8.3.1. Allowed Parent Dependencies

- JavaServiceWrapper

6.8.4. Attributes

6.8.4.1. Exported Attributes

Name	Property
`javaServiceWrapperAppParameters`	settingValue

6.9. JavaServiceWrapperConsoleTitle

6.9.1. Overview

JavaServiceWrapperConsoleTitle: *Title to use when running the Java Service Wrapper as a console*

If configured to run in a console window, this will be the title given to the window.

See: wrapper.console.title Property.

6.9.2. Design

Super Type
JavaServiceWrapperSetting

Role	**Concrete**. (Objects can be created.)
Instance Names	**Unique**

6.9.3. Constraints

6.9.3.1. Allowed Parent Dependencies

- JavaServiceWrapper

6.9.4. Attributes

6.9.4.1. Exported Attributes

Name	Property
`javaServiceWrapperConsoleTitle`	settingValue

6.10. JavaServiceWrapperNtServiceName

6.10.1. Overview

JavaServiceWrapperNtServiceName: *Identifying Name of the Java Service Wrapper Windows service*

This is the identifier used by Windows, and should be unique among all Windows Service. Make sure that all JavaServiceWrapper objects have different values for this setting.

See: wrapper.ntservice.name Property.

6.10.2. Design

Super Type
JavaServiceWrapperSetting

Role	**Concrete**. (Objects can be created.)
Instance Names	**Unique**

6.10.3. Constraints

6.10.3.1. Allowed Parent Dependencies

- JavaServiceWrapper

6.10.4. Attributes

6.10.4.1. Exported Attributes

Name	Property	Description
javaServiceWrapperNtServ	settingValue	This attribute is used by the lifecycle commands of JavaServiceWrapper to uniquely identify the Windows Service that is being managed.

6.11. JavaServiceWrapperNtServiceDisplayName

6.11.1. Overview

JavaServiceWrapperNtServiceDisplayName: *Displayed name of the Java Service Wrapper Windows service*

See: wrapper.ntservice.displayname Property.

6.11.2. Design

Super Type
JavaServiceWrapperSetting

Role	**Concrete**. (Objects can be created.)
Instance Names	**Unique**

6.11.3. Constraints

6.11.3.1. Allowed Parent Dependencies

- JavaServiceWrapper

6.11.4. Attributes

6.11.4.1. Exported Attributes

Name	Property
javaServiceWrapperNtServiceDisplayNam	settingValue

6.12. JavaServiceWrapperNtServiceDescription

6.12.1. Overview

JavaServiceWrapperNtServiceDescription: *Description of the Java Service Wrapper Windows service*

See: wrapper.ntservice.description Property.

6.12.2. Design

Super Type
JavaServiceWrapperSetting

Role	**Concrete**. (Objects can be created.)
Instance Names	**Unique**

6.12.3. Constraints

6.12.3.1. Allowed Parent Dependencies

- JavaServiceWrapper

6.12.4. Attributes

6.12.4.1. Exported Attributes

Name	Property
`javaServiceWrapperNtServiceDescriptic`	settingValue

6.13. JavaServiceWrapperNtServiceStartType

6.13.1. Overview

JavaServiceWrapperNtServiceStartType: *Mode in which the Java Service Wrapper Windows service is installed. AUTO_START or DEMAND_START.*

See: wrapper.ntservice.starttype Property.

6.13.2. Design

Super Type
JavaServiceWrapperSetting

Role	**Concrete**. (Objects can be created.)

Instance Names	**Unique**

6.13.3. Constraints

6.13.3.1. Allowed Parent Dependencies

- JavaServiceWrapper

6.13.3.2. Allowed Property Values

Property	Allowed Values	Default	Enforced
settingValue	• `AUTO_START` • `DEMAND_START`	• `AUTO_START`	true

6.13.4. Attributes

6.13.4.1. Exported Attributes

Name	Property
`javaServiceWrapperNtServiceStartType`	settingValue

6.14. JavaServiceWrapperNtServiceInteractive

6.14.1. Overview

JavaServiceWrapperNtServiceInteractive: *Whether or not to allow the Java Service Wrapper Windows service to interact with the desktop. (true/false)*

See: wrapper.ntservice.interactive Property.

6.14.2. Design

Super Type
JavaServiceWrapperSetting

Role	**Concrete**. (Objects can be created.)
Instance Names	**Unique**

6.14.3. Constraints

6.14.3.1. Allowed Parent Dependencies

- JavaServiceWrapper

6.14.3.2. Allowed Property Values

Property	Allowed Values	Default	Enforced
settingValue	• false • true	• true	true

6.14.4. Attributes

6.14.4.1. Exported Attributes

Name	Property
javaServiceWrapperNtServiceInteractiv	settingValue

6.15. JavaServiceWrapperNtServiceAccount

6.15.1. Overview

JavaServiceWrapperNtServiceAccount: *Account to use with the Java Service Wrapper Windows service.*

Note:
The usage of this setting is not currently enabled in this implementation.

See: wrapper.ntservice.account Property.

6.15.2. Design

Super Type
JavaServiceWrapperSetting

Role	**Concrete**. (Objects can be created.)
Instance Names	**Unique**

6.15.3. Constraints

6.15.3.1. Allowed Parent Dependencies

- JavaServiceWrapper

6.15.4. Attributes

6.15.4.1. Exported Attributes

Name	Property
`javaServiceWrapperNtServiceAccount`	settingValue

6.16. JavaServiceWrapperNtServicePassword

6.16.1. Overview

JavaServiceWrapperNtServicePassword: *Password to use with the Java Service Wrapper Windows service.*

> **Note:**
> The usage of this setting is not currently enabled in this implementation.

See: wrapper.ntservice.password Property.

6.16.2. Design

Super Type
JavaServiceWrapperSetting

Role	**Concrete**. (Objects can be created.)
Instance Names	**Unique**

6.16.3. Constraints

6.16.3.1. Allowed Parent Dependencies

- JavaServiceWrapper

6.16.4. Attributes

6.16.4.1. Exported Attributes

Name	Property
`javaServiceWrapperNtServicePassword`	settingValue