

# AtgAppServerCollection

## A collection of ATG Servers

### Table of contents

1 Overview.....	3
2 Design.....	3
3 Constraints.....	3
3.1 Allowed Child Dependencies.....	3
3.2 Allowed Parent Dependencies.....	3
4 Commands.....	3
4.1 Prepare.....	3
4.2 Install.....	4
4.3 Update.....	4
4.4 Packages-Install.....	5
4.5 Status.....	5
4.6 Configure.....	5
4.7 Start.....	6
4.8 Stop.....	6
4.9 Restart.....	6
4.10 dispatchCmd.....	7
4.11 switchDatastore.....	7
4.12 packagesInstallChildren.....	8
4.13 packagesInstallSelf.....	8
4.14 updateDatastoreModelObject.....	8
4.15 invalidateCache.....	9
4.16 mkdirStructure.....	9
4.17 generateInstallUnit.....	9

4.18 getDatastoreName.....	10
5 Related Types.....	10
5.1 AtgAppServerCollectionSetting.....	10
5.2 AtgEnvironment.....	11
5.3 AtgDynamoInstallUnit.....	11
5.4 AtgServerPattern.....	12

## 1. Overview

[Open in Workbench](#)

**AtgAppServerCollection:** *A collection of ATG Servers*

## 2. Design

**Super Type**  
[ServiceMediator](#)

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

## 3. Constraints

### 3.1. Allowed Child Dependencies

- [AtgJBossServer](#)

### 3.2. Allowed Parent Dependencies

- [AtgSite](#)
- Node

## 4. 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.

### 4.1. Prepare

*prepares and validates install*

**Usage**

Prepare

**4.1.1. Workflow****4.2. Install***Installs self and local child resources***Usage**

```
Install [-basedir <${entity.instance.dir}>] [-depot
<${context.depot}>] [-name <${context.name}>] [-nodir]
[-nodispatch] [-nomodule] [-noproperties] [-subdirs
<bin,conf,logs,var>] [-type <${context.type}>]
```

**4.2.1. Options**

Option	Description
basedir	<i>base directory</i>
depot	<i>project depot</i>
name	<i>object name</i>
nodir	<i>do not create directories</i>
nodispatch	<i>do not dispatch Install action to child resources</i>
nomodule	<i>do not update module</i>
noproperties	<i>do not update properties</i>
subdirs	<i>directory list</i>
type	<i>object type</i>

**4.3. Update***performs a software deployment***Usage**

```
Update [-command <Update>]
```

**4.3.1. Workflow**

1. [dispatchCmd](#)

### 4.3.2. Options

Option	Description
command	<i>command to dispatch</i>

### 4.4. Packages-Install

*installs all the package dependencies*

#### Usage

Packages-Install [-command <Packages-Install>]

#### 4.4.1. Workflow

1. [dispatchCmd](#)

#### 4.4.2. Options

Option	Description
command	<i>command to dispatch</i>

### 4.5. Status

*Check services*

#### Usage

Status [-command <Status>]

#### 4.5.1. Workflow

1. [dispatchCmd](#)

#### 4.5.2. Options

Option	Description
command	<i>command to dispatch</i>

### 4.6. Configure

*Configure services*

#### Usage

Configure [-command <Configure>]

#### 4.6.1. Workflow

1. [dispatchCmd](#)

#### 4.6.2. Options

Option	Description
command	<i>command to dispatch</i>

#### 4.7. Start

*Start services*

##### Usage

Start [-command <Start>]

#### 4.7.1. Workflow

1. [dispatchCmd](#)

#### 4.7.2. Options

Option	Description
command	<i>command to dispatch</i>

#### 4.8. Stop

*Stop services*

##### Usage

Stop [-command <Stop>]

#### 4.8.1. Workflow

1. [dispatchCmd](#)

#### 4.8.2. Options

Option	Description
command	<i>command to dispatch</i>

#### 4.9. Restart

*Restart services*

**Usage**

```
Restart [-command <Restart>]
```

**4.9.1. Workflow**1. [dispatchCmd](#)**4.9.2. Options**

Option	Description
command	<i>command to dispatch</i>

**4.10. dispatchCmd**

*dispatches command to child*

**Usage**

```
dispatchCmd [-buildstamp <>] -command <> [-keepgoing]
[-resourcename <.*>] [-resourceorder <[^.]*>] [-resourcetype
<[^.]*>] [-sortorder <ascending>] [-threadcount <6>]
```

**4.10.1. Options**

Option	Description
buildstamp	<i>Unique build and deployment identifier</i>
command	<i>command to dispatch</i>
keepgoing	<i>If true, all iterations of the called workflow will be executed, even if a task in one or more of them fails.</i>
resourcename	<i>resource name pattern</i>
resourceorder	<i>resource order name</i>
resourcetype	<i>resource type name</i>
sortorder	<i>order to sort resources</i>
threadcount	<i>Number of concurrent threads to dispatch</i>

**4.11. switchDatastore**

*switches the datastore*

**Usage**

```
switchDatastore [-atgservertype <>] -datastore <>
```

**4.11.1. Options**

Option	Description
atgservertype	<i>type name</i>
datastore	<i>datastore name</i>

**4.12. packagesInstallChildren**

*Workflow runs packagesInstall process*

**Usage**

```
packagesInstallChildren [-command <Packages-Install>]
```

**4.12.1. Workflow**

1. [dispatchCmd](#)

**4.12.2. Options**

Option	Description
command	<i>command to dispatch</i>

**4.13. packagesInstallSelf**

*installs packages*

**Usage**

```
packagesInstallSelf [-packagetype <[^\.]*>]
```

**4.13.1. Options**

Option	Description
packagetype	<i>type of packages to install</i>

**4.14. updateDatastoreModelObject**

*updates the datastore model object*

**Usage**

```
updateDatastoreModelObject [-datastore <>]
```

**4.14.1. Options**

Option	Description
datastore	<i>datastore name</i>

**4.15. invalidateCache**

*invalidates the caches*

**Usage**

invalidateCache [-atgservertype <>]

**4.15.1. Options**

Option	Description
atgservertype	<i>atg server type name</i>

**4.16. mkdirStructure**

*creates directories and makes necessary symlinks*

**Usage**

mkdirStructure [-dynamo\_root <>] [-name <>]

Execution	bash
Arguments	echo implement procedure

**4.16.1. Options**

Option	Description
dynamo_root	<i>dynamo install dir</i>
name	<i>install unit name</i>

**4.17. generateInstallUnit**

*Creates the app install unit*

**Usage**

generateInstallUnit [-dynamo\_root <>] [-name <>]

**4.17.1. Options**

Option	Description
dynamo_root	<i>dynamo root directory</i>
name	<i>name of the instal unit</i>

## 4.18. getDatastoreName

*gets the current datastore name*

### Usage

```
getDatastoreName [-atgservertype <>]
```

### 4.18.1. Options

Option	Description
atgservertype	<i>type pattern</i>

## 5. Related Types

The following types are defined for use with AtgAppServerCollection.

### 5.1. AtgAppServerCollectionSetting

#### 5.1.1. Overview

[Open in Workbench](#)

**AtgAppServerCollectionSetting:** *an AtgAppServerCollection setting*

#### 5.1.2. Design

##### Super Type Setting

Role	<b>Abstract.</b> (Objects cannot be created.)
Instance Names	<b>Unique</b>

#### 5.1.3. Constraints

##### 5.1.3.1. Allowed Parent Dependencies

- [AtgAppServerCollection](#)

## 5.2. AtgEnvironment

### 5.2.1. Overview

[Open in Workbench](#)

**AtgEnvironment:** *An Atg environment*

### 5.2.2. Design

#### Super Type

[AtgAppServerCollectionSetting](#)

Role	<b>Concrete.</b> (Objects can be created.)
Instance Names	<b>Unique</b>

### 5.2.3. Attributes

#### 5.2.3.1. Exported Attributes

Name	Property
environment	settingValue

## 5.3. AtgDynamoInstallUnit

### 5.3.1. Overview

[Open in Workbench](#)

**AtgDynamoInstallUnit:** *ATG install unit name*

### 5.3.2. Design

#### Super Type

[AtgAppServerCollectionSetting](#)

Role	<b>Concrete.</b> (Objects can be created.)
Instance Names	<b>Unique</b>

### 5.3.3. Attributes

#### 5.3.3.1. Exported Attributes

Name	Property
atgInstallUnit	settingValue

## 5.4. AtgServerPattern

### 5.4.1. Overview

[Open in Workbench](#)

**AtgServerPattern:** *Type name or regex to dispatch commands*

### 5.4.2. Design

#### Super Type

[AtgAppServerCollectionSetting](#)

Role	<b>Concrete.</b> (Objects can be created.)
Instance Names	<b>Unique</b>

### 5.4.3. Attributes

#### 5.4.3.1. Exported Attributes

Name	Property
atgServerPattern	settingValue