


Space Details

Key:	CARGO
Name:	Cargo
Description:	Uniform J2EE Container Control System
Creator (Creation Date):	bwalding (Aug 14, 2004)
Last Modifier (Mod. Date):	bwalding (Aug 14, 2004)



Available Pages

- Home 
- Navigation
- Tested on
- SVN
- Using from Java
- Using from Ant
- Credits































Cargo provides a Java API to start/stop and configure Java containers

Status

Version status:

Version	Status	Comments
0.1		Released on 11/09/04
0.2		To be released in October 2004

Container implementation status:

Container	Java API/version	Ant API/version	Maven API/version
JBoss 3.x	 0.2	 0.2	 N/A
Jetty 4.x	 0.1	 0.2	 N/A
Orion 1.x	 0.1	 0.1	 N/A
Orion 2.x	 0.1	 0.1	 N/A
Resin 2.x	 0.1	 0.1	 N/A
Resin 3.x	 0.1	 0.1	 N/A
Tomcat 3.x	 0.1	 0.1	 N/A
Tomcat 4.x	 0.1	 0.1	 N/A
Tomcat 5.x	 0.1	 0.1	 N/A
WebLogic 7.x	 0.2	 0.2	 N/A

As glitches may happen even after a container is released for the first time, e.g. if a new feature is added to the framework, but not supported by all containers, we encourage you to report your success/failures in the [Tested on](#) section.

Feature list

- Provides a Java API to:

- Start containers
- Stop containers
- Configure containers for deployment in any user-specified directory
- Wait for containers to be started
- Wait for containers to be stopped
- Supports WAR and EAR static deployments
- Provides Ant tasks that wraps the Java API.

In addition the Cargo project also offers a Java API to manipulate J2EE descriptors (currently `web.xml` and `application.xml`). Most notably the API allows merging two `web.xml` files.

Quick Start

The following piece of code demonstrates how to configure Resin 3.0.8 to start in `target/resin3x` and deploy a WAR located in `src/testinput/simple.war`. The default port is 8080. Please note that the `container.start()` and `container.stop()` methods wait until the container is fully started and fully stopped before continuing. Thus, for any action you are executing after, you are assured the container is completely operational.

WARNING: Do not point the working dir to an existing directory. Everything in this directory will be deleted. This is a dangerous bug in version 0.1 that will be fixed in version 0.2

```
Container container = new Resin3xContainer();
container.setHomeDir("c:/apps/resin-3.0.8");
container.setWorkingDir("target/resin3x");

Deployable deployable = new WAR("src/testinput/simple.war");

container.addDeployable(deployable);

container.start();

// At this point you are assured the container is started.

container.stop();

// At this point you are assured the container is stopped.
```

Navigation

This page last changed on Sep 11, 2004 by [vmassol](#).

Cargo

- [Home](#)
- [Using from Java](#)
- [Using from Ant](#)
- [Javadoc](#)
- [License](#)

Download

- [Cargo 0.1](#)

Support

- [Issues](#)
- [Roadmap](#)
- [Change log](#)

Community

- [Mailing Lists](#)
- [Who we are](#)

Developers

- [SVN](#)
- [Wiki](#)
- [Maven site](#)
- [Credits](#)

Tested on

This page last changed on Sep 19, 2004 by [vmassol](#).

In this section you can find the test status of the different containers for the different Cargo releases.

This page will then contain results of testing the framework in real world configurations.

Add your own experiences to the section matching your framework version, using the following format:

- Tomcat
 - 4.1.27 (J2EE 1.2 and J2EE 1.3) – [Vincent Massol](#), on 10th of September 3001
 - 4.1.28 (J2EE 1.3) **failed** – jerome@coffebreaks.org, on 11th of September 3001

Cargo 0.2

- Resin
- Tomcat
- Orion
- Jetty

Cargo 0.1

- Resin
 - 3.0.8 (J2EE 1.3) – [Vincent Massol](#), on 17 Aug 2004
- Tomcat
 - 3.3.2 (J2EE 1.3) – [Vincent Massol](#), on 5 Sep 2004
 - 4.1.30 (J2EE 1.3) – [Vincent Massol](#), on 5 Sep 2004
 - 5.0.25 (J2EE 1.3) – [Vincent Massol](#), on 5 Sep 2004
 - 5.0.28 (J2EE 1.3) – [Vincent Massol](#), on 5 Sep 2004
- Orion
 - 1.6.0b (J2EE 1.3) – [Vincent Massol](#), on 17 Aug 2004
 - 2.0.3 (J2EE 1.3) – [Vincent Massol](#), on 17 Aug 2004
- Jetty
 - 4.1.20 (J2EE 1.3) – [Vincent Massol](#), on 18 Aug 2004
 - 4.2.17 (J2EE 1.3) – [Vincent Massol](#), on 19 Sep 2004

SVN

This page last changed on Aug 20, 2004 by [vmassol](#).

For general information see the [SVN page on Codehaus](#).

Web Access

<http://svn.cargo.codehaus.org>

Anonymous SVN Access

```
svn co svn://svn.cargo.codehaus.org/cargo/scm/cargo/trunk
```

Developer SVN Access via SSH

```
svn co svn+ssh://svn.cargo.codehaus.org/home/projects/cargo/scm/cargo/trunk
```

SVN Access behind a firewall

Currently Codehaus does not support WebDAV access.

Examples using the Cargo Java API

WARNING: Do not point the working dir to an existing directory. Everything in this directory will be deleted. This is a dangerous bug in version 0.1 that will be fixed in version 0.2

Starting Resin 3.x with no deployables

Note: The `homeDir` and `workingDir` property are mandatory.

```
Container container = new Resin3xContainer();
container.setHomeDir("c:/apps/resin-3.0.8");
container.setWorkingDir("target/resin3x");
container.start();
```

Starting Orion 2.x with an EAR to deploy

```
Container container = new Orion2xContainer();
container.setHomeDir("c:/apps/orion-2.0.3");
container.setWorkingDir("target/orion2x");

EAR ear = new EAR("src/data/some.ear");
container.addDeployable(ear);

container.start();
```

Starting Jetty 4.x with a WAR to deploy

Note: Unlike the other containers, the Jetty integration does not require the Jetty container to be installed. You simply need to add the Jetty jar (`org.mortbay.jetty.jar`), the Servlet API jar (`javax.servlet.jar`), and the Tomcat Jasper jars (`jasper-compiler.jar`, `jasper-runtime.jar`) to your classpath. Thus the `homeDir` property has not effect.

```
Container container = new Jetty4xContainer();
container.setWorkingDir("target/jetty4x");

WAR war = new WAR("src/data/some.war");
container.addDeployable(war);

container.start();
```

Starting Tomcat 4.x specifying an output console log file

```
Container container = new Tomcat4xContainer();
container.setHomeDir("c:/apps/jakarta-tomcat-4.1.30");
container.setWorkingDir("target/tomcat4x");

container.setOutput("target/output.log");

container.start();
```

Use the `container.setAppend(true|false)` method to decide whether the log file is recreated or whether it is appended to, keeping the previous execution logs.

Starting Tomcat 5.x on a specific port

```
Container container = new Tomcat5xContainer();
container.setHomeDir("c:/apps/jakarta-tomcat-5.0.25");
container.setWorkingDir("target/tomcat5x");

container.setPort(8888);

container.start();
```

Starting Orion 1.x with some additional classpath entries

This can be useful if you need to add some jars to the container classpath. For example if you have instrumented your source code with Clover you'll need to add the Clover jar to the classpath.

```
Container container = new Orion1xContainer();
container.setHomeDir("c:/apps/orion-1.6.0b");
container.setWorkingDir("target/orion1x");

container.setExtraClasspath(new String[] { "libs/clover.jar" });

container.start();
```

Starting Tomcat 3.x with some System properties set in the container JVM

```
Container container = new Tomcat3xContainer();
container.setHomeDir("c:/apps/jakarta-tomcat-3.3.2");
container.setWorkingDir("target/tomcat3x");
```



```
Map props = new HashMap();
props.put("myproperty", "myvalue");
container.setSystemProperties(props);

container.start();
```

Examples using Cargo with Ant

WARNING: Do not point the working dir to an existing directory. Everything in this directory will be deleted. This is a dangerous bug in version 0.1 that will be fixed in version 0.2

Before being able to use the Cargo tasks you need to register them against Ant. This is done by using the Ant `<taskdef>` element:

```
<taskdef resource="cargo.tasks">
  <classpath>
    <pathelement location="{cargo.jar}"/>
  </classpath>
</taskdef>
```

Starting Resin 3.x with no deployables

Note: The `homeDir` and `workingDir` properties are mandatory.

```
<cargo-resin3x homeDir="c:/apps/resin-3.0.8" workingDir="target/resin3x"
action="start"/>
```

Starting Orion 2.x with an EAR to deploy

```
<cargo-orion2x homeDir="c:/apps/orion-2.0.3" workingDir="target/orion2x"
action="start">
  <ear earFile="src/data/some.ear"/>
</cargo-orion2x>
```

Starting Orion 1.x with a WAR to deploy

```
<cargo-orion1x homeDir="c:/apps/orion-1.6.0b" workingDir="target/orion1x"
action="start">
  <war warFile="src/data/some.war"/>
</cargo-orion1x>
```

Starting Tomcat 4.x specifying an output console log file

```
<cargo-tomcat4x homeDir="c:/apps/jakarta-tomcat-4.1.30" workingDir="target/tomcat4x"
output="target/output.log" action="start"/>
```

Use the `append="true|false"` attribute to decide whether the log file is recreated or whether it is appended to, keeping the previous execution logs.

Starting Tomcat 5.x on a specific port

```
<cargo-tomcat5x homeDir="c:/apps/jakarta-tomcat-5.0.25" workingDir="target/tomcat5x"
port="8888" action="start"/>
```

Starting Orion 1.x with some additional classpath entries

This can be useful if you need to add some jars to the container classpath. For example if you have instrumented your source code with Clover you'll need to add the Clover jar to the classpath.

```
<cargo-orion1x homeDir="c:/apps/orion-1.6.0b" workingDir="target/orion1x"
action="start">
  <extraClasspath>
    <pathelement location="libs/clover.jar"/>
  </extraClasspath>
</cargo-orion1x>
```

Starting Tomcat 3.x with some System properties set in the container JVM

```
<cargo-tomcat3x homeDir="c:/apps/jakarta-tomcat-3.3.2" workingDir="target/tomcat3x"
action="start">
  <sysproperty key="myproperty" value="myvalue"/>
</cargo-tomcat3x>
```

Credits

This page last changed on Sep 04, 2004 by [vmassol](#).

The following persons deserve credit for Cargo:

- Apache and The Jakarta cactus project: Cargo started as a refactoring of the Cactus Ant integration subproject
- [Vincent Massol](#): Lead developer of Cargo (and of Cactus)
- Christopher Lenz: Has developed most of the Cactus Ant integration code that has eventually found its way in Cargo
- Desire Atanga: Implementatoin of Tomcat support for the Java API
- Jerome Lacoste: General ideas and discussions about Cargo