


## Space Details

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

### Available Pages

- Home 
- Navigation
- Tested on
- SVN
- Credits
- Documentation Archives
- News
- Test Results
- Features
  - Ant support
  - Classpath configuration
  - Configuration
  - Configuration properties
  - Container Factory
  - Container instance creation
  - Debugging
  - Deployable
  - Deployment descriptor API
  - Embedded mode
  - Existing configuration
  - Installer
  - Maven support
  - Passing system properties
  - Standalone configuration
  - Standalone mode
  - Start
  - Static deployment of EAR
  - Static deployment of expanded WAR
  - Static deployment of WAR
  - Stop
- Containers
  - JBoss 3.x

- Jetty 4.x
- Oc4J 9.x
- Orion 1.x
- Orion 2.x
- Resin 2.x
- Resin 3.x
- Tomcat 3.x
- Tomcat 4.x
- Tomcat 5.x
- WebLogic 8.x
- Release procedure

## Mission





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

Possible use cases for Cargo:

- To start containers for integration and functional tests
- To start containers for applications that require a container to be started (Plugins for IDEs, etc)

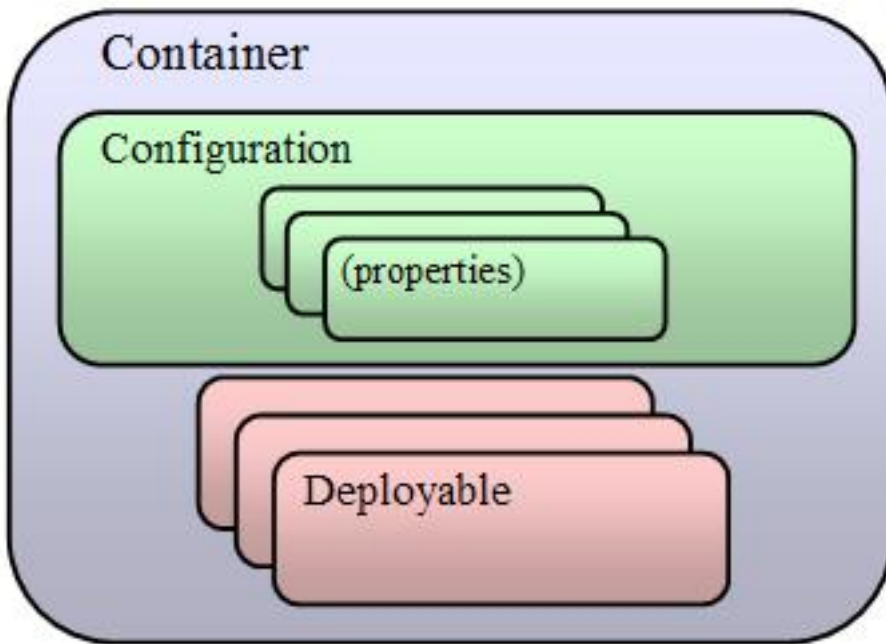
## Status

**Version status** (click in the status column to get release notes):

Version	Status	Comments
<a href="#">0.1</a>		Released on 11/09/04
<a href="#">0.2</a>		Released on 03/10/04
<a href="#">0.3</a>		Released on 30/10/04
<a href="#">0.4</a>		Released on 26/11/04

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.

## Architecture




































- The [Container](#) is the top level interface wrapping a real physical container. It is composed of a [Configuration](#) and of [Deployables](#).
- A [Configuration](#) tells Cargo how the container is to be configured (whether it should create a standalone setup, whether it should be based on an existing configuration, etc)
- [Deployables](#) are archives to be deployed in the [Container](#). They are WARs, EARs, etc.

## Feature list

- [Ant support](#) — Cargo provides Ant tasks to perform all the operations available from the Java API
- [Classpath configuration](#) — How to configure Cargo's classpath
- [Configuration](#) — Specifies how the container is configured
- [Configuration properties](#) — Properties to configure a container (request port, shutdown port, logging level, threads, etc)
- [Container Factory](#) — Instantiate a container by name
- [Container instance creation](#) — Create a container instance
- [Debugging](#) — Explain how to perform debugging when something doesn't work in Cargo
- [Deployable](#) — Deployables are archives (WAR, EAR, etc) that can be deployed in the container
- [Deployment descriptor API](#) — API to manipulate J2EE descriptors (currently `web.xml` and `application.xml`)
- [Embedded mode](#)
- [Existing configuration](#) — Not yet implemented
- [Installer](#) — Installs a container

- [Maven support](#) — Not yet implemented
- [Passing system properties](#) — How to pass system properties that will be available to the container while executing
- [Standalone configuration](#) — Configures your container in a specific directory
- [Standalone mode](#)
- [Start](#) — Start a container that is not already running
- [Static deployment of EAR](#) — Deploy an EAR that will be started when the container starts
- [Static deployment of expanded WAR](#)
- [Static deployment of WAR](#) — Deploy a WAR that will be started when the container starts
- [Stop](#) — Stop a running container

## Container support

Container	Java API/version	Ant API/version	Maven API/version
<a href="#">JBoss 3.x</a>	 ???	 ???	 N/A
<a href="#">Jetty 4.x</a>	 0.1	 ???	 N/A
<a href="#">OC4J 9.x</a>	 0.3	 0.3	 N/A
<a href="#">Orion 1.x</a>	 0.1	 0.1	 N/A
<a href="#">Orion 2.x</a>	 0.1	 0.1	 N/A
<a href="#">Resin 2.x</a>	 0.1	 0.1	 N/A
<a href="#">Resin 3.x</a>	 0.1	 0.1	 N/A
<a href="#">Tomcat 3.x</a>	 0.1	 0.1	 N/A
<a href="#">Tomcat 4.x</a>	 0.1	 0.1	 N/A
<a href="#">Tomcat 5.x</a>	 0.1	 0.1	 N/A
<a href="#">WebLogic 8.x</a>	 0.3	 0.3	 N/A

## 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.

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

Deployable deployable =
    container.getDeployableFactory().createWAR("path/to/simple.war");
container.addDeployable(deployable);

container.start();

// Here you are assured the container is started.

container.stop();

// Here you are assured the container is stopped.
```

## Navigation

---

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

## Cargo 0.4 doc

- [Home](#)
- [News](#)
- [Features](#)
- [Javadoc](#)
- [license](#)

## Archives

- [Doc Archives](#)

## Download

- [Cargo 0.3](#)
- [Cargo 0.3 doc](#)

## Containers

- [JBoss 3.x](#)
- [Jetty 4.x](#)
- [Orion 1.x](#)
- [Orion 2.x](#)
- [Resin 2.x](#)
- [Resin 3.x](#)
- [Tomcat 3.x](#)
- [Tomcat 4.x](#)
- [Tomcat 5.x](#)
- [Weblogic 8.x](#)
- [Oc4j 9.x](#)

## Support

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

## Community

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

## Developers

- [Credits](#)
- [SVN](#)
- [Wiki](#)
- [Maven site](#)
- [Release procedure](#)
- [DC status](#)





## Maven support

---

This page last changed on Oct 24, 2004 by [vmassol](#).

### Definition

Not yet implemented

## Tested on

This page last changed on Nov 06, 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](#)
  - 4.1.28 (J2EE 1.3) **failed** - [jerome@coffeebreaks.org](mailto:jerome@coffeebreaks.org)

### Cargo 0.4

<a href="#">jira.codehaus.org</a> (3 issues)			
Summary	Assignee	Reporter	Status
<a href="#">Test on Orion 2.0.5</a>	Vincent Massol	Vincent Massol	 Closed
<a href="#">Test on Tomcat 4.1.31</a>	Vincent Massol	Vincent Massol	 Closed
<a href="#">Test on Orion 2.0.4</a>	Vincent Massol	Vincent Massol	 Closed

### Cargo 0.3

<a href="#">jira.codehaus.org</a> (4 issues)			
Summary	Assignee	Reporter	Status
<a href="#">Test on WebLogic 8.1 SP3</a>	Vincent Massol	Vincent Massol	 Closed
<a href="#">Test on OC4J 9.0.4</a>	Vincent Massol	Vincent Massol	 Closed
<a href="#">Test on Tomcat 5.5.3-snapshot</a>	Vincent Massol	Vincent Massol	 Closed
<a href="#">Test on Resin 3.0.9</a>	Vincent Massol	Vincent Massol	 Closed

## Cargo 0.2

- Resin
  - 3.0.8 (J2EE 1.3) – [Vincent Massol](#)
  - 3.0.9 (J2EE 1.3) **failed** – [Vincent Massol](#)
    - It fails because Resin 3.0.9 no longer supports the <cache> directive (it's now only support for the professional versions). See [CARGO-44](#). Support added in Cargo 0.3.
- Tomcat
  - 3.3.2 (J2EE 1.3) – [Vincent Massol](#)
  - 4.1.30 (J2EE 1.3) – [Vincent Massol](#)
  - 5.0.25 (J2EE 1.3) – [Vincent Massol](#)
  - 5.0.28 (J2EE 1.3) – [Vincent Massol](#)
  - 5.5.2 (J2EE 1.3) – [Vincent Massol](#)
  - 5.5.3-alpha (J2EE 1.3) – [Vincent Massol](#)
- Orion
- Jetty

## Cargo 0.1

- Resin
  - 3.0.8 (J2EE 1.3) – [Vincent Massol](#)
- Tomcat
  - 3.3.2 (J2EE 1.3) – [Vincent Massol](#)
  - 4.1.30 (J2EE 1.3) – [Vincent Massol](#)
  - 5.0.25 (J2EE 1.3) – [Vincent Massol](#)
  - 5.0.28 (J2EE 1.3) – [Vincent Massol](#)
- Orion
  - 1.6.0b (J2EE 1.3) – [Vincent Massol](#)
  - 2.0.3 (J2EE 1.3) – [Vincent Massol](#)
- Jetty
  - 4.1.20 (J2EE 1.3) – [Vincent Massol](#)
  - 4.2.17 (J2EE 1.3) – [Vincent Massol](#)

## 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.

## Credits

---

This page last changed on Nov 01, 2004 by [vmassol](#).

The following persons deserve credit for Cargo.

Special thanks:

- **Apache and The Jakarta cactus project:** Cargo started as a refactoring of the [Cactus](#) Ant integration subproject
- **Christopher Lenz:** Has developed most of the Cactus Ant integration code that spawned the Cargo project

Committers:

- [Vincent Massol](#): Lead developer of Cargo (and of Cactus)
- [Desire ATANGA](#): Implementation of Tomcat and WebLogic support

Code contributors:

- [Jerome Lacoste](#): General ideas and discussions about Cargo
- [Tim Shadel](#): Implementation of OC4J support
- [Arnaud HERITIER](#): Several Maven related patches to improve the Cargo build
- Matt Raible: Asked for improvements to the Tomcat support so that Cargo can support nicely AppFuse. Provided patches to improve Tomcat support.

A special mention to the following Cargo users who have helped make Cargo better and helped promote it:

- Mike-Cannon Brookes: Asked for expanded war support

If we have forgotten anyone, please accept our apologies and feel free to mention on the list so that we can correct the error.

## Documentation Archives

---

This page last changed on Oct 30, 2004 by [vmassol](#).

This web site contains the documentation for the next version of Cargo.

Available documentation PDFs:

- [Cargo 0.1 documentation](#)
- [Cargo 0.2 documentation](#)
- [Cargo 0.3 documentation](#)

## Supported Features

Feature category	Feature name	Supported	Comments
<b>Java API</b>	<a href="#">Start</a>		
	<a href="#">Stop</a>		
	<a href="#">Extra classpath passing</a>		
	<a href="#">Container factory</a>		
	<a href="#">Debugging</a>		
	<a href="#">Passing system properties</a>		
	<a href="#">Standalone configuration</a>		
	<a href="#">Existing configuration</a>		
	<a href="#">Static deployment of WAR</a>		
	<a href="#">Static deployment of expanded WAR</a>		
	<a href="#">Static deployment of EAR</a>		
	<a href="#">Standalone mode</a>		
	<a href="#">Embedded mode</a>		
<b>Ant API</b>	<a href="#">Ant support</a>		
<b>Maven API</b>	<a href="#">Maven support</a>		
<b>Properties</b>	<a href="#">ServletPropertySet.PC</a>		
	<a href="#">GeneralPropertySet.H</a>		TNAME
	<a href="#">GeneralPropertySet.L</a>		GING

## Instantiating in Java

```
Container container = new Resin3xContainer();  
[...]
```

## Instantiating in Ant

```
<cargo-resin3x [...]  
</cargo-resin3x>
```



## Supported Features

Feature category	Feature name	Supported	Comments
<b>Java API</b>	<a href="#">Start</a>		
	<a href="#">Stop</a>		
	<a href="#">Extra classpath passing</a>		
	<a href="#">Container factory</a>		
	<a href="#">Debugging</a>		
	<a href="#">Passing system properties</a>		
	<a href="#">Standalone configuration</a>		
	<a href="#">Existing configuration</a>		
	<a href="#">Static deployment of WAR</a>		
	<a href="#">Static deployment of expanded WAR</a>		
	<a href="#">Static deployment of EAR</a>		
	<a href="#">Standalone mode</a>		
	<a href="#">Embedded mode</a>		
<b>Ant API</b>	<a href="#">Ant support</a>		
<b>Maven API</b>	<a href="#">Maven support</a>		
<b>Properties</b>	<a href="#">ServletPropertySet.PC</a>		
	<a href="#">GeneralPropertySet.H</a>		TNAME
	<a href="#">GeneralPropertySet.L</a>		GING

## Instantiating in Java

```
Container container = new Resin2xContainer();  
[...]
```

## Instantiating in Ant

```
<cargo-resin2x [...]  
</cargo-resin2x>
```

## Supported Features

Feature category	Feature name	Supported	Comments
<b>Java API</b>	<a href="#">Start</a>		
	<a href="#">Stop</a>		
	<a href="#">Extra classpath passing</a>		
	<a href="#">Container factory</a>		
	<a href="#">Debugging</a>		
	<a href="#">Passing system properties</a>		
	<a href="#">Standalone configuration</a>		
	<a href="#">Existing configuration</a>		
	<a href="#">Static deployment of WAR</a>		
	<a href="#">Static deployment of expanded WAR</a>		
	<a href="#">Static deployment of EAR</a>		
	<a href="#">Standalone mode</a>		
	<a href="#">Embedded mode</a>		
	<b>Ant API</b>	<a href="#">Ant support</a>	
<b>Maven API</b>	<a href="#">Maven support</a>		
<b>Properties</b>	<a href="#">ServletPropertySet.PC</a>		
	<a href="#">GeneralPropertySet.H</a>	 TNAME	
	<a href="#">GeneralPropertySet.L</a>	 GING	

Custom configuration properties:

Property name	Java constant	Valid values	Description	Example
---------------	---------------	--------------	-------------	---------

	to use			
cargo.orion.rmi.port	OrionPropertySetting	RMI_PORT	Port for the Orion RMI server	"25791"

## Instantiating in Java

```
Container container = new Orion2xContainer();
[...]
```

## Instantiating in Ant

```
<cargo-orion2x [...]
</cargo-orion2x>
```

## Supported Features

Feature category	Feature name	Supported	Comments
<b>Java API</b>	<a href="#">Start</a>		
	<a href="#">Stop</a>		
	<a href="#">Extra classpath passing</a>		
	<a href="#">Container factory</a>		
	<a href="#">Debugging</a>		
	<a href="#">Passing system properties</a>		
	<a href="#">Standalone configuration</a>		
	<a href="#">Existing configuration</a>		
	<a href="#">Static deployment of WAR</a>		
	<a href="#">Static deployment of expanded WAR</a>		
	<a href="#">Static deployment of EAR</a>		
	<a href="#">Standalone mode</a>		
	<a href="#">Embedded mode</a>		
	<b>Ant API</b>	<a href="#">Ant support</a>	
<b>Maven API</b>	<a href="#">Maven support</a>		
<b>Properties</b>	<a href="#">ServletPropertySet.PC</a>		
	<a href="#">GeneralPropertySet.H</a>	 TNAME	
	<a href="#">GeneralPropertySet.L</a>	 GING	

Custom configuration properties:

Property name	Java constant	Valid values	Description	Example
---------------	---------------	--------------	-------------	---------

	to use			
cargo.orion.rmi.port	OrionPropertySet	RMI_PORT	Port for the Orion RMI server	"25791"

## Instantiating in Java

```
Container container = new OrionlxContainer();
[...]
```

## Instantiating in Ant

```
<cargo-orionlx [...]
</cargo-orionlx>
```

## Supported Features

Feature category	Feature name	Supported	Comments
<b>Java API</b>	<a href="#">Start</a>		
	<a href="#">Stop</a>		
	<a href="#">Extra classpath passing</a>		
	<a href="#">Container factory</a>		
	<a href="#">Debugging</a>		
	<a href="#">Passing system properties</a>		
	<a href="#">Standalone configuration</a>		
	<a href="#">Existing configuration</a>		
	<a href="#">Static deployment of WAR</a>		
	<a href="#">Static deployment of expanded WAR</a>		
	<a href="#">Static deployment of EAR</a>		
	<a href="#">Standalone mode</a>		
	<a href="#">Embedded mode</a>		
	<b>Ant API</b>	<a href="#">Ant support</a>	
<b>Maven API</b>	<a href="#">Maven support</a>		
<b>Properties</b>	<a href="#">ServletPropertySet.PC</a>		
	<a href="#">GeneralPropertySet.H</a>	 TNAME	
	<a href="#">GeneralPropertySet.L</a>	 GING	

## Instantiating in Java












```
Container container = new Tomcat3xContainer();  
[...]
```

## Instantiating in Ant

```
<cargo-tomcat3x [...]  
</cargo-tomcat3x>
```



## Supported Features

Feature category	Feature name	Supported	Comments
Java API	<a href="#">Start</a>		
	<a href="#">Stop</a>		
	<a href="#">Extra classpath passing</a>		
	<a href="#">Container factory</a>		
	<a href="#">Debugging</a>		
	<a href="#">Passing system properties</a>		
	<a href="#">Standalone configuration</a>		
	<a href="#">Existing configuration</a>		
	<a href="#">Static deployment of WAR</a>		Does not support META-INF/context.xml files yet
	<a href="#">Static deployment of expanded WAR</a>		
	<a href="#">Static deployment of EAR</a>		
	<a href="#">Standalone mode</a>		
	<a href="#">Embedded mode</a>		
Ant API	<a href="#">Ant support</a>		
Maven API	<a href="#">Maven support</a>		
Properties	<a href="#">ServletPropertySet.PC</a>		
	<a href="#">GeneralPropertySet.H</a>		TNAME
	<a href="#">GeneralPropertySet.L</a>		GING

Custom configuration properties:

Property name	Java constant to use	Valid values	Description	Example
cargo.tomcat.shutdown	TomcatPropertySet.SHUTDOWN_PORT	1-65535	TCP/IP port number on which this server waits for a shutdown command	"8205"

## Instantiating in Java

```
Container container = new Tomcat4xContainer();
[...]
```

## Instantiating in Ant

```
<cargo-tomcat4x [...]
</cargo-tomcat4x>
```

## Tomcat 5.x

This page last changed on Oct 23, 2004 by vmassol.

### Supported Features

**Note: Tomcat 5.5.x is supported (Requires JDK 1.5+)**

Feature category	Feature name	Supported	Comments
<b>Java API</b>	<a href="#">Start</a>		
	<a href="#">Stop</a>		
	<a href="#">Extra classpath passing</a>		
	<a href="#">Container factory</a>		
	<a href="#">Debugging</a>		
	<a href="#">Passing system properties</a>		
	<a href="#">Standalone configuration</a>		
	<a href="#">Existing configuration</a>		
	<a href="#">Static deployment of WAR</a>		
	<a href="#">Static deployment of expanded WAR</a>		
	<a href="#">Static deployment of EAR</a>		
	<a href="#">Standalone mode</a>		
	<a href="#">Embedded mode</a>		
<b>Ant API</b>	<a href="#">Ant support</a>		
<b>Maven API</b>	<a href="#">Maven support</a>		
<b>Properties</b>	<a href="#">ServletPropertySet.PC</a>		
	<a href="#">GeneralPropertySet.H</a>		TNAME
	<a href="#">GeneralPropertySet.L</a>		GING

Custom configuration properties:

Property name	Java constant to use	Valid values	Description	Example
cargo.tomcat.shutdown	TomcatPropertySet.SHUTDOWN_PORT	1-65535	TCP/IP port number on which this server waits for a shutdown command	"8205"

## Instantiating in Java

```
Container container = new Tomcat5xContainer();
[...]
```

## Instantiating in Ant

```
<cargo-tomcat5x [...]
</cargo-tomcat5x>
```

## Supported Features

Feature category	Feature name	Supported	Comments
<b>Java API</b>	<a href="#">Start</a>		
	<a href="#">Stop</a>		
	<a href="#">Extra classpath passing</a>		
	<a href="#">Container factory</a>		
	<a href="#">Debugging</a>		
	<a href="#">Passing system properties</a>		
	<a href="#">Standalone configuration</a>		
	<a href="#">Existing configuration</a>		
	<a href="#">Static deployment of WAR</a>		
	<a href="#">Static deployment of expanded WAR</a>		
	<a href="#">Static deployment of EAR</a>		
	<a href="#">Standalone mode</a>		
	<a href="#">Embedded mode</a>		
<b>Ant API</b>	<a href="#">Ant support</a>		
<b>Maven API</b>	<a href="#">Maven support</a>		
<b>Properties</b>	<a href="#">ServletPropertySet.PC</a>		
	<a href="#">GeneralPropertySet.H</a>		TNAME
	<a href="#">GeneralPropertySet.L</a>		GING

## Instantiating in Java

```
Container container = new Jetty4xEmbeddedContainer();  
[...]
```

## Supported Features

**Note: Not implemented yet**

Feature category	Feature name	Supported	Comments
<b>Java API</b>	<a href="#">Start</a>		
	<a href="#">Stop</a>		
	<a href="#">Extra classpath passing</a>		
	<a href="#">Container factory</a>		
	<a href="#">Debugging</a>		
	<a href="#">Passing system properties</a>		
	<a href="#">Standalone configuration</a>		
	<a href="#">Existing configuration</a>		
	<a href="#">Static deployment of WAR</a>		
	<a href="#">Static deployment of expanded WAR</a>		
	<a href="#">Static deployment of EAR</a>		
	<a href="#">Standalone mode</a>		
	<a href="#">Embedded mode</a>		
<b>Ant API</b>	<a href="#">Ant support</a>		
<b>Maven API</b>	<a href="#">Maven support</a>		
<b>Properties</b>	<a href="#">ServletPropertySet.PC</a>		
	<a href="#">GeneralPropertySet.H</a> TNAME		
	<a href="#">GeneralPropertySet.L</a> GING		

## Instantiating in Java

```
Container container = new JBoss3x();  
[...]
```

## Instantiating in Ant

```
<cargo-jboss3x [...]  
</cargo-jboss3x>
```



## Debugging

---

This page last changed on Oct 23, 2004 by [vmassol](#).

### Definition

Explain how to perform debugging when something doesn't work in Cargo. Indeed it can happen that the container does not start or stop as expected. Or that some deployable does not deploy fine. Or whatever else! Here is a short list of things you can do to try debugging the problem.

### Redirecting container output to a file

The `container.setOutput(File)` API allows you redirect the container console (stdout) to a file. This is the first file you should check in case of problem.

### Example using the Java API

Starting Tomcat 4.x specifying an output console log file:

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

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.

### Example using the Ant API

Starting Tomcat 4.x specifying an output console log file:

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

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.

### Generating Cargo logs

Some Cargo classes support generation of logs. This is implemented through the

notion of [Monitor](#).

For example to turn on logging monitoring on a `Container` class, you can use:

```
Monitor fileMonitor = new FileMonitor(new File("c:/tmp/cargo.log"), true);
container.setMonitor(fileMonitor);
```

There are several Monitors that are readily available in the Cargo distribution:

- [FileMonitor](#): logs messages to a file
- [SimpleMonitor](#): logs messages to the console (stdout)

## Turning on container logs

Cargo is able to configure containers to generate various levels logs. There are 3 levels defined: "low", "medium" and "high". They represent the quantity of information you wish in the generated log file. You can turn on container logging by using the following API:

```
container.setProperty(GeneralPropertySet.LOGGING, "medium");
```

The generated log files will then be found in the Working directory you have specified on the container (through the `container.setWorkingDir()` call).

When using the Ant tasks, you can specify the log file by using the `log` attribute. For example:

```
<cargo-resin3x [...] log="target/cargo.log"/>
```

## News

---

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

**[Blog: Cargo \(CARGO\)](#) (RSS 0.91)**

(Uniform J2EE Container Control System)






**[Cargo 0.4 released](#)**


Release 0.4 is now available on <http://cargo.codehaus.org>

Download: <http://dist.codehaus.org/cargo/jars/cargo-0.4.jar>

Release Notes: (<http://tinyurl.com/3t7g3>)

**[jira.codehaus.org](#) (10 issues)**

T	Key	Summary	Assignee	Reporter
	<a href="#">CARGO-88</a>	<a href="#">Test on Orion 2.0.5</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-87</a>	<a href="#">Test on Tomcat 4.1.31</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-81</a>	<a href="#">Create ConfigurationFactory and merge notions of Configuration and ContainerConfiguration</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-78</a>	<a href="#">Test on Orion 2.0.4</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-77</a>	<a href="#">Add new &lt;syspropertyset file=""&gt; nested element in the Cargo Ant task</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-76</a>	<a href="#">Add Container.getTimeout() API</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-74</a>	<a href="#">Add support for container keys in the &lt;cargo&gt; Ant task</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-73</a>	<a href="#">Orion changed</a>	Vincent Massol	Arnaud

		<a href="#">the download URL</a>		HERITIER
	<a href="#">CARGO-72</a>	<a href="#">Introduce the notion of "id/idref" in the Cargo Ant tasks</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-20</a>	<a href="#">Fix build pb when running master build for first time</a>	Vincent Massol	Vincent Massol


### [How to use Cargo for functional testing](#)









Vincent Massol has posted an article describing [how to use Cargo for functional testing](#) on his [blog](#).










### [Cargo 0.3 released](#)




Release 0.3 is now available on <http://cargo.codehaus.org>  
Download: <http://dist.codehaus.org/cargo/jars/cargo-0.3.jar>  
Release Notes: (<http://tinyurl.com/7xbea>)

### [jira.codehaus.org](#) (25 issues)

T	Key	Summary	Assignee	Reporter
	<a href="#">CARGO-84</a>	<a href="#">Test on WebLogic 8.1 SP3</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-83</a>	<a href="#">Test on OC4J 9.0.4</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-82</a>	<a href="#">Test on Tomcat 5.5.3-snapshot</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-79</a>	<a href="#">Test on Resin 3.0.9</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-70</a>	<a href="#">Failing to run Tomcat 4.x+ when WAR specified using a</a>	Vincent Massol	Matt Raible

		<a href="#">relative path</a>		
	<a href="#">CARGO-69</a>	<a href="#">The goal cargo:clover launched in the root directory fails</a>	Vincent Massol	Arnaud HERITIER
	<a href="#">CARGO-67</a>	<a href="#">Complete web site reorganization</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-62</a>	<a href="#">Tomcat 4.x - Loading files from WEB-INF directory doesn't work</a>	Vincent Massol	Matt Raible
	<a href="#">CARGO-61</a>	<a href="#">In the ZipURLInstaller fall back to the target directory if offline</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-60</a>	<a href="#">Add a Container.getState() API to get the container state</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-58</a>	<a href="#">Add a "log" attribute to the Cargo tasks to provide a file where to put Cargo logs</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-57</a>	<a href="#">Use a default Tomcat shutdown port different than 8005</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-56</a>	<a href="#">Add property to configure RMI server port of</a>	Vincent Massol	Vincent Massol

		<a href="#">Orion containers</a>		
	<a href="#">CARGO-55</a>	<a href="#">Fix the nasty way of stopping Resin 3.x that we have (thread killing)</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-53</a>	<a href="#">Jetty4xEmbedded container fails some sample tests</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-52</a>	<a href="#">Introduce notion of Configuration</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-51</a>	<a href="#">Executing tests with several containers in a row fail the test sometimes</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-50</a>	<a href="#">Add proxy support to the ZipURLInstaller class</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-49</a>	<a href="#">Adding customization of port attribute for Catalina server config.xml</a>	Vincent Massol	Anatol Pomozov
	<a href="#">CARGO-48</a>	<a href="#">Document the Installer/ZipURLInstaller feature</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-46</a>	<a href="#">Add continuous build for Cargo</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-45</a>	<a href="#">Add notion of Installer to automatically install a container</a>	Vincent Massol	Vincent Massol

		<a href="#">distribution</a>		
	<a href="#">CARGO-44</a>	<a href="#">Add support for Resin 3.0.9</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-43</a>	<a href="#">Add support for Tomcat 5.5</a>	Vincent Massol	Matt Raible
	<a href="#">CARGO-41</a>	<a href="#">Add support for WebLogic 8.x in the Java and Ant APIs</a>	Desire ATANGA	Vincent Massol

### **[Cargo to be featured in Spring Live](#)**

[Matt Raible has announced the Cargo 0.2 release on his blog](#) and plans to feature Cargo in his upcoming [Spring Live](#) book:

I've been looking for something like Cargo for a while now - mainly so I could provide an easy way to test JSPs (and therefore the whole app) in-container. For the last couple of years, I've been using Cactus' <runservertests> task, but it requires you to configure your own startup and shutdown targets - which can be difficult for the different containers. Cargo makes this easy. So easy that I've added it as a topic to Chapter 8 of Spring Live. BTW, the MyUsers sample app shows you how to use jMock and Easy Mock for isolating Manager tests and Action/Controller tests - something that I might eventually move to with AppFuse.

Good work Matt!

### **[Cargo 0.2 released](#)**











Release 0.2 is now available on <http://cargo.codehaus.org>







Download: <http://dist.codehaus.org/cargo/jars/cargo-0.2.jar>

Release Notes: (<http://tinyurl.com/4db8o>)



[jira.codehaus.org](http://jira.codehaus.org) (16 issues)

T	Key	Summary	Assignee	Reporter
	<a href="#">CARGO-40</a>	<a href="#">Add DeployableFactory class + Container.getDeployable() API to support container specific deployable implementations</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-39</a>	<a href="#">Add support for Tomcat's META-INF/context.xml</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-38</a>	<a href="#">Add new Container.isStarted() API</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-37</a>	<a href="#">cargocpc does not work with Servlet 2.2</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-35</a>	<a href="#">Create a News page on the web site</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-33</a>	<a href="#">Add Debugging page on web site</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-31</a>	<a href="#">Refactor Jetty4xContainer into Jetty4xEmbeddedContainer</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-30</a>	<a href="#">Add web site doc for ContainerFactory</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-29</a>	<a href="#">Add support for expanded WARs</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-28</a>	<a href="#">Added</a>	Vincent Massol	Vincent Massol

		<a href="#">verification of ServletPropertySet.PORT property for valid value</a>		
	<a href="#">CARGO-27</a>	<a href="#">Cargo deleted all my source</a>	Vincent Massol	Mike Cannon-Brookes
	<a href="#">CARGO-26</a>	<a href="#">Add container debugging properties</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-24</a>	<a href="#">Add new Container.getId() API</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-23</a>	<a href="#">Increase container timeout period</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-22</a>	<a href="#">Implement a generic solution for container properties</a>	Vincent Massol	Vincent Massol
	<a href="#">CARGO-21</a>	<a href="#">Add ContainerFactory class</a>	Vincent Massol	Vincent Massol

### **[Matt Raible talks about Cargo](#)**

[http://raibledesigns.com/page/rd?anchor=ann\\_cargo\\_0\\_1\\_released](http://raibledesigns.com/page/rd?anchor=ann_cargo_0_1_released)

### **[Cargo 0.1 on TheServerSide](#)**

[Thread on Cargo 0.1 on TSS.](#)

### **[Cargo 0.1 released](#)**

Release 0.1 is now available on <http://cargo.codehaus.org>

This initial version 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
- Ant tasks that wraps the Java API.

The Java API supports the following containers:

- Resin 2.x
- Resin 3.x
- Orion 1.x
- Orion 2.x
- Tomcat 3.x
- Tomcat 4.x
- Tomcat 5.x
- Jetty 4.x

The Ant tasks support all of the above except the Jetty 4.x container.

All documentation is available on: <http://cargo.codehaus.org>

Many thanks to (more details on <http://cargo.codehaus.org/Credits>):

- Desire Atanga for the Tomcat support
- Jerome Lacoste for general discussions

### **Cargo is born**

CCI has been renamed Cargo and has found a [home](#) on Codehaus.

### **Container Client Interface on TheServerSide**

[Article and discussion thread on CCI](#). See also [Vincent's blog on CCI](#).

## Supported Features

Feature category	Feature name	Supported	Comments
<b>Java API</b>	<a href="#">Start</a>		
	<a href="#">Stop</a>		
	<a href="#">Extra classpath passing</a>		
	<a href="#">Container factory</a>		
	<a href="#">Debugging</a>		
	<a href="#">Passing system properties</a>		
	<a href="#">Standalone configuration</a>		
	<a href="#">Existing configuration</a>		
	<a href="#">Static deployment of WAR</a>		
	<a href="#">Static deployment of expanded WAR</a>		
	<a href="#">Static deployment of EAR</a>		
	<a href="#">Standalone mode</a>		
	<a href="#">Embedded mode</a>		
	<b>Ant API</b>	<a href="#">Ant support</a>	
<b>Maven API</b>	<a href="#">Maven support</a>		
<b>Properties</b>	<a href="#">ServletPropertySet.PC</a>		
	<a href="#">GeneralPropertySet.H</a>		TNAME
	<a href="#">GeneralPropertySet.L</a>		GING

Custom configuration properties:

Property name	Java constant	Valid values	Description	Example
---------------	---------------	--------------	-------------	---------

	to use			
cargo.orion.rmi.port	OptionPropertySetting	RMI_PORT	Port for the Orion RMI server	"25791"

## Instantiating in Java

```
Container container = new Oc4j9xContainer();
[...]
```

## Instantiating in Ant

```
<cargo-oc4j9x [... ]
</cargo-oc4j9x>
```

## Test Results

---

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

Core Java API unit tests:

Ant API unit tests:

Samples/Java functional tests:

Samples/Ant functional tests:

## Features

---

This page last changed on Oct 22, 2004 by [vmassol](#).

- [Ant support](#) — Cargo provides Ant tasks to perform all the operations available from the Java API
- [Classpath configuration](#) — How to configure Cargo's classpath
- [Configuration](#) — Specifies how the container is configured
- [Configuration properties](#) — Properties to configure a container (request port, shutdown port, logging level, threads, etc)
- [Container Factory](#) — Instantiate a container by name
- [Container instance creation](#) — Create a container instance
- [Debugging](#) — Explain how to perform debugging when something doesn't work in Cargo
- [Deployable](#) — Deployables are archives (WAR, EAR, etc) that can be deployed in the container
- [Deployment descriptor API](#) — API to manipulate J2EE descriptors (currently `web.xml` and `application.xml`)
- [Embedded mode](#)
- [Existing configuration](#) — Not yet implemented
- [Installer](#) — Installs a container
- [Maven support](#) — Not yet implemented
- [Passing system properties](#) — How to pass system properties that will be available to the container while executing
- [Standalone configuration](#) — Configures your container in a specific directory
- [Standalone mode](#)
- [Start](#) — Start a container that is not already running
- [Static deployment of EAR](#) — Deploy an EAR that will be started when the container starts
- [Static deployment of expanded WAR](#)
- [Static deployment of WAR](#) — Deploy a WAR that will be started when the container starts
- [Stop](#) — Stop a running container

## Start

---

This page last changed on Oct 23, 2004 by [vmassol](#).

### Definition

Start a container that is not already running

### Explanation

First you need to create a `Container` instance. This can be done using the [container factory](#) or directly by instating a [container](#) implementation class.

Once you have this container instance, starting the container is as simple as calling the `start()` method. Before doing this though you'll need to ensure you have defined the container's `homeDir` (if you're using a container in [standalone mode](#) - It's not required for containers in [embedded mode](#)).

Of course if you wish to statically deploy archives, you'll need to add [deployables](#) to the container.

It is important to note that the `Container.start()` method will wait until the container is **fully started** before returning.

### Example using the Java API

Starting Resin 3.x with no deployable:

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

container.start();
```

### Example using the Ant API

Before being able to use the Cargo Ant tasks you need to register them against Ant. This is done by using the Ant `<taskdef>` element. See the [Ant support page](#). The action to start the container is specified using the `action="start"` attribute as shown below.



Starting Resin 3.x with no deployable:

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

## Stop

---

This page last changed on Oct 23, 2004 by [vmassol](#).

### Definition

Stop a running container

Note: The stop action waits till the container is fully stopped before returning.

### Example using the Java API

Stopping Orion 1.x:

```
Container container = new OrionlxContainer();
container.setHomeDir("c:/apps/orion-1.6.0b");

container.stop();
```

### Example using the Ant API

Stopping Orion 1.x:

```
<cargo-orionlx homeDir="c:/apps/orion-1.6.0b" action="stop"/>
```

## Standalone configuration

This page last changed on Nov 12, 2004 by [vmassol](#).

### Definition

Configures your container in a specific directory

### Explanation

The [standalone configuration](#) allows configuring your container so that it is setup to start in a directory you choose (see the [configuration page](#) for more general explanations).

There are 2 ways of using a standalone configuration:

- By directly instantiating the configuration matching your container. For example:

```
[...]
Configuration configuration = new CatalinaStandaloneConfiguration(container,
"target/tomcat5x");
container.setConfiguration(configuration);
[...]
```

- By using the `ConfigurationFactory` which automatically maps the right implementation for the container you're using. For example:

```
[...]
ConfigurationFactory factory = new ConfigurationFactory();
Configuration configuration = factory.createConfiguration(container,
ConfigurationFactory.STANDALONE, "target/tomcat5x");
container.setConfiguration(configuration);
[...]
```

Note that if you don't specify any [configuration](#), Cargo will use a [standalone configuration](#) by default and the target directory will point to a directory named after your container's id in your user's temporary directory.

### Example using the Ant API

```
<cargo-tomcat5x [...]>
  <configuration hint="standalone" dir="target/tomcat5x"/>
  [...]
</cargo-tomcat5x >
```

```
</cargo-tomcat5x>
```

## Existing configuration

---

This page last changed on Oct 24, 2004 by [vmassol](#).

### Definition

Not yet implemented

## Installer

---

This page last changed on Oct 23, 2004 by [vmassol](#).

### Definition

Installs a container

### Explanation

An Installer is meant to install a container. There is currently only a single Installer implementation: `ZipURLInstaller` which downloads a zipped container distribution from a URL and which installs it (i.e. unpacks it) in a specified directory. This is useful if you wish to fully automate a container installation without having to ask the user to manually install a container on their machine.

### Example

```
Installer installer = new ZipURLInstaller(
    "http://www.caucho.com/download/resin-3.0.9.zip",
    "target/installs");
installer.install();

Container container = new Resin3xContainer();
container.setHomeDir(installer.getHomeDir());
[...]
```

## Static deployment of WAR

---

This page last changed on Oct 22, 2004 by [vmassol](#).

### Definition

Deploy a WAR that will be started when the container starts

### Example

Let's see how to Jetty 4.x (in [embedded mode](#)) with a WAR to deploy in it.

*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 (`servletapi.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 Jetty4xEmbeddedContainer();

Deployable war = container.getDeployableFactory().createWAR("src/data/some.war");
container.addDeployable(war);

container.start();
```

## Static deployment of expanded WAR

---

This page last changed on Oct 22, 2004 by [vmassol](#).



## Static deployment of EAR

---

This page last changed on Oct 23, 2004 by [vmassol](#).

### Definition

Deploy an EAR that will be started when the container starts

### Example using the Java API

Starting Orion 2.x with an EAR to deploy:

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

Deployable ear = container.getDeployableFactory().createEAR("src/data/some.ear");
container.addDeployable(ear);

container.start();
```

### Example using the Ant API

Starting Orion 2.x with an EAR to deploy:

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

## Standalone mode

---

This page last changed on Oct 22, 2004 by [vmassol](#).

## Embedded mode

---

This page last changed on Oct 22, 2004 by [vmassol](#).

Cargo provides different container implementations. A Container implementation can be either [standalone](#) or [embedded](#). The embedded mode means that Cargo is using directly the container's Java API to control it. If you're using one of the embedded implementation you'll need to ensure that you have the container's jars in your classpath.

Advantages of embedded mode:

- Faster. There's no need to start a new JVM nor new threads.
- Simpler. There's no need to install the container in a directory

Here is the list of container implementations that support the embedded mode:

- `Jetty4xEmbeddedContainer`: [Jetty 4.x](#) implementation

## Configuration properties

This page last changed on Nov 12, 2004 by [vmassol](#).

### Definition

Properties to configure a container (request port, shutdown port, logging level, threads, etc)

### Explanations

It is possible to set container configuration properties using the Cargo API. These properties are applied to a [Configuration](#).

There are 2 kinds of properties:

- General properties
- Container-specific properties. See each [container](#)'s page for a list of the custom properties it supports.

General properties:

Property name	Java constant to use	Valid values	Description	Example
cargo.servlet.port	ServletPropertyServletPort	Integer	Port on which the Servlet/JSP container will listen to	"8280"
cargo.hostname	GeneralPropertyServletHost	String HOSTNAME	Host name on which the container will listen to	"myserver"
cargo.logging	GeneralPropertyServletLevel	String "low", "medium" or "high"	Level representing the quantity of information we wish to log	"medium"

### Example using the Java API

Starting Tomcat 5.x on a specific port:

```
Container container = new Tomcat5xContainer();
container.setHomeDir("c:/apps/jakarta-tomcat-5.0.29");
Configuration configuration = new CatalinaStandaloneConfiguration(container);

configuration.setProperty(ServletPropertySet.PORT, "8081");

container.setConfiguration(configuration);
[...]
```

## Example using the Ant API

Starting Tomcat 5.x on a specific port:

```
<cargo-tomcat5x homeDir="c:/apps/jakarta-tomcat-5.0.29" action="start">
  <configuration>
    <property name="cargo.servlet.port" value="8081"/>
  </configuration>
</cargo-tomcat5x>
```

## Container Factory

---

This page last changed on Oct 22, 2004 by [vmassol](#).

### Definition

Instantiate a container by name

### Explanation

There are 2 solutions to instantiate a container:

- by explicitly creating a new instance of the container itself. For example to instantiate a Resin 3.x container:

```
Container container = new Resin3xContainer();
```

- by using the `ContainerFactory` class. The advantage is then that you can instantiate by name and thus your code can be generic which is nice if you plan to run the same code with different containers. For example, to instantiate a Resin 3.x container:

```
ContainerFactory factory = new ContainerFactory();  
Container container = factory.createContainer("resin3x");
```

Note: You can also pass the full container class name (that's useful if you wish to instantiate a custom container you have developed):

```
Container container =  
factory.createContainer("org.codehaus.cargo.container.resin.Resin3xContainer");
```

## Containers

---

This page last changed on Oct 22, 2004 by [vmassol](#).

Here is the list of supported containers.

- [JBoss 3.x](#)
- [Jetty 4.x](#)
- [Oc4J 9.x](#)
- [Orion 1.x](#)
- [Orion 2.x](#)
- [Resin 2.x](#)
- [Resin 3.x](#)
- [Tomcat 3.x](#)
- [Tomcat 4.x](#)
- [Tomcat 5.x](#)
- [WebLogic 8.x](#)

## Configuration

---

This page last changed on Nov 12, 2004 by [vmassol](#).

### Definition

Specifies how the container is configured

### Explanation

The notion of Configuration is different from the notion of Installation. Indeed a container is installed in the place where you have installed it (or where the [Cargo Installer](#) has installed it).

However, it is possible to configure this container to tell it deploy archives to some other directories and consider this other directory as the new home directory of the container. All containers support this feature. This allows to leave an installed container directory intact (no modifications) and to compartment all modifications to some directory that you are controlling. Say you wish to automate some functional tests. You can set the new container home to point to your temporary build directory. This is the configuration we call [Standalone configuration](#).

By opposition if you choose to use the default configuration that your container has set up when you installed it, it is called an [Existing configuration](#).

Please note that at the moment Cargo only implements [Standalone configurations](#) for all the supported containers. However, you can create your own implementation if you wish (see below).

### Custom configuration

The Cargo API allow the user to plug his/her own custom configuration implementation. The only requisites is that the implementation must implements `ContainerConfiguration` and have 2 constructors: `MyCustomConfiguration(Container)` and `MyCustomConfiguration(Container, File homeDir)`.

You can then register your configuration against the `ConfigurationFactory` class. It's optional and only required if you want to let users use `ConfigurationFactory` to instantiate your configuration. You would write:



```
ConfigurationFactory factory = new ConfigurationFactory();
factory.registerConfiguration("containerIdOfAssociatedContainer", "customhint",
MyCustomConfiguration.class);
```

Where `customhint` is any value you wish, used to differentiate your configuration from other existing ones associated with the container represented by its id.

## Deployable

This page last changed on Oct 23, 2004 by [vmassol](#).

### Definition

Deployables are archives (WAR, EAR, etc) that can be deployed in the container

### Explanation

A `Deployable` class is a wrapper class around a physical archive. `Deployable`s are constructed using a `DeployableFactory` provided by your container. The reason for this factory is to support container extensions to archives (for example, Tomcat supports `context.xml` files located in your WAR's `META-INF` directory, JBoss allows for a `jboss-web.xml` located in your WAR, etc).

The `DeployableFactory` interface offers different methods for creating `Deployable`s (e.g. `DeployableFactory.createEAR(String)`, `DeployableFactory.createWAR(String)`, etc).

Once you have a `Deployable` instance wrapping your archive, you can tell the Container to deploy it when the container starts. This is achieved by calling the `Container.addDeployable(Deployable)` API. You can also read [how to statically deploy a WAR](#) or [how to statically deploy an EAR](#).

In the near future we'll add support to deploy `Deployable`s in a running container (a.k.a. dynamic deployments).

### Example using the Java API

Deploying a WAR in Tomcat 5.x:

```
Container container = new Tomcat5xContainer();
container.setHomeDir("c:/apps/tomcat-5.0.29");

DeployableFactory factory = container.getDeployableFactory();
WAR war = factory.createWAR("path/to/my.war");

container.addDeployable(war);

container.start();
```

### Example using the Ant API

## Deploying a WAR in Tomcat 5.x:

```
<cargo-tomcat5x homeDir="c:/apps/tomcat-5.0.29" action="start">  
  <war warfile="path/to/my.war" />  
</cargo-tomcat5x>
```

### Definition

Cargo provides Ant tasks to perform all the operations available from the Java API

### Explanation

Before using the Ant API you need to register the Cargo Ant tasks into Ant. This is done in the following manner:

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

### Example

Here's a full example showing how to deploy a WAR, and expanded WAR and an EAR in an Orion 2.x container. Please note that the `output` and `log` attribute are optional. The `property` elements allow you to tune how the container is configured. Here we're telling it to start on port 8180 and to generate the maximum amount of logs in the container output file.

```
<taskdef resource="cargo.tasks">
  <classpath>
    <pathelement location="path/to/cargo.jar"/>
  </classpath>
</taskdef>

<cargo-orion2x homeDir="c:/apps/orion-2.0.3" output="target/output.log"
  log="target/cargo.log" action="start">
  <war warfile="path/to/my/simple.war"/>
  <war warfile="path/to/my/expandedwar/simple"/>
  <ear earfile="path/to/my/simple.ear"/>
  <property name="cargo.servlet.port" value="8180"/>
  <property name="cargo.logging" value="high"/>
</cargo-orion2x>
```

## Deployment descriptor API

---

This page last changed on Oct 22, 2004 by [vmassol](#).

### Definition

API to manipulate J2EE descriptors (currently `web.xml` and `application.xml`)

### Explanation

TODO. Most notably the API allows merging two `web.xml` files.

### Example

TODO

## Container instance creation

---

This page last changed on Oct 22, 2004 by [vmassol](#).

### Definition

Create a container instance

### Explanation

A container instance is created by simply instantiating the Java object implementing the container. Each container implementation offers a main Java object wrapping its container and which allows to manipulate the container (start, stop, configure, deploy archives, etc).

The class to use for instantiating a container can be found in each container's page:

- [JBoss 3.x](#)
- [Jetty 4.x](#)
- [Oc4J 9.x](#)
- [Orion 1.x](#)
- [Orion 2.x](#)
- [Resin 2.x](#)
- [Resin 3.x](#)
- [Tomcat 3.x](#)
- [Tomcat 4.x](#)
- [Tomcat 5.x](#)
- [WebLogic 8.x](#)

In addition it's possible to [instantiate a container by name](#).

### Example

```
Container container = new Orion2xContainer();
[...]
```

```
Container container = new Resin3xContainer();
[...]
```

```
Container container = new Weblogic8xContainer();
[...]
```

## Classpath configuration

---

This page last changed on Oct 23, 2004 by [vmassol](#).

### Definition

How to configure Cargo's classpath

### Explanation

TODO (explain the 2 modes: standalone and embedded and how Cargo treats both WRT classpath. Explain also the `setExtraClasspath()` API)

### Example using the Java API

Starting Orion 1.x with Clover jar added to its 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.setExtraClasspath(new String[] { "libs/clover.jar" });

container.start();
```

### Example using the Ant API

Starting Orion 1.x with some additional classpath entries:

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

## Passing system properties

---

This page last changed on Oct 23, 2004 by [vmassol](#).

### Definition

How to pass system properties that will be available to the container while executing

### Explanations

TODO

### Example using the Java API

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");

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

container.start();
```

### Example using the Ant API

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

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



## Supported Features

Feature category	Feature name	Supported	Comments
<b>Java API</b>	<a href="#">Start</a>		
	<a href="#">Stop</a>		
	<a href="#">Extra classpath passing</a>		
	<a href="#">Container factory</a>		
	<a href="#">Debugging</a>		
	<a href="#">Passing system properties</a>		
	<a href="#">Standalone configuration</a>		
	<a href="#">Existing configuration</a>		
	<a href="#">Static deployment of WAR</a>		
	<a href="#">Static deployment of expanded WAR</a>		
	<a href="#">Static deployment of EAR</a>		
	<a href="#">Standalone mode</a>		
	<a href="#">Embedded mode</a>		
<b>Ant API</b>	<a href="#">Ant support</a>		
<b>Maven API</b>	<a href="#">Maven support</a>		
<b>Properties</b>	<a href="#">ServletPropertySet.PC</a>		
	<a href="#">GeneralPropertySet.H</a>	 TNAME	
	<a href="#">GeneralPropertySet.L</a>	 GING	

## Instantiating in Java

```
Container container = new WebLogic8xContainer();  
[...]
```

## Instantiating in Ant

```
<cargo-weblogic8x [...]  
</cargo-weblogic8x>
```

## Release procedure

---

This page last changed on Nov 26, 2004 by [vmassol](#).

1. Perform a clean SVN checkout
2. Edit top level `project.xml` and modify `<currentVersion>` tag. Ex: from `0.3-SNAPSHOT` to `0.3`
3. Run `maven` at top level to generate the distribution
4. Run `maven cargo:site` to generate the development site. Check the result. If ok, run `maven cargo:site-deploy` to deploy the site to `beaver.codehaus.org`. You'll need the following properties set up in your `build.properties` (replace with your own data):
  - `maven.username = vmassol`
  - `maven.ssh.executable = tortoiseplink`
  - `maven.scp.executable = pscp`
5. Upload the Cargo jar by running `maven cargo:deploy`. You'll need the following properties set up in your `build.properties` (replace with your own data):
  - `maven.repo.list = codehaus`
  - `maven.repo.codehaus.username = vmassol`
  - `maven.repo.codehaus.privatekey = C:/Documents and Settings/Vincent Massol/My Documents/.ssh/vmassol.ssh2`
  - `maven.repo.codehaus.passphrase =`
6. Log onto Cargo JIRA, release the current version and add the next version
7. Check that the Cargo wiki is up to date. Specifically, perform the following updates:
  - a. modify the status on the home page about the delivery
  - b. export the wiki to PDF and add the generated PDF to the docs archive page
  - c. modify the navigation page to include the latest download link and docs
8. Tag SVN by copying the HEAD to `svn+ssh://svn.cargo.codehaus.org/home/projects/cargo/scm/cargo/tags/<version>`
9. Send an announcement email to Cargo dev and Cargo user mailing lists (and to any other onsite site, magazines, etc)
10. Create a [blog post](#)
11. Modify again `project.xml` and modify `<currentVersion>` for the next version