

Container Start

Definition

Start a container that is not already running

i This feature is only available for [local containers](#)

Explanation

First you need to create a [container instance](#).

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 `home` (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 `LocalContainer.start()` method will wait until the container is **fully started** before returning.

Example using the Java API

Starting Resin 3.x with no deployable:

```
InstalledLocalContainer container = new
Resin3xInstalledLocalContainer(
    new
Resin3xStandaloneLocalConfiguration("target/
resin3x"));
container.setHome("c:/apps/resin-3.0.15");

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 containerId="resin3x"  
home="c:/apps/resin-3.0.15" action="start"/>
```

Example using the Maven2/Maven3 plugin

For the Maven2/Maven plugin, please read: [Starting and stopping a container](#).

Other tips

Letting the started container outlive CARGO's process

By default, the container started by CARGO is linked to CARGO's process; which means that once the Java process that has started the container is finished (be it via Java API, ANT or the Maven2/Maven3 plugin) the started container will also be killed.

In some cases, mostly if you want to use CARGO as a "launcher script", you need the started container to "outlive" CARGO's process, i.e. that the started container keeps running even after CARGO itself has terminated. This can be achieved by simply setting the property `GeneralPropertySet.SPAWN_PROCESS` (or, in the ANT tasks or Maven2/Maven3 plugin, the `cargo.process.spawn` property) to `true`.

Note that this feature is only available for standalone containers (i.e., not for embedded containers).