

# Deployer

## Definition

Performs a hot deployment of a [Deployable](#)

## Explanation

You use a `Deployer` when you wish to deploy a [Deployable](#) into a running container (this is known as [Hot Deployment](#)). There are 2 types of Deployers:

- [Local Deployer](#): A local deployer deploys in a locally installed container. Usually, local deployers use the file system to perform the deployment by copying the deployable to a container-specific target directory.
- [Remote Deployer](#): A remote deployer is used to deploy to a container that can be on the same machine or on some remote machine. Deploying remotely requires passing information such as username and password of the user to use for deploying, etc. These information are passed using a [Runtime Configuration](#).

## Deployer features

- [Local Deployer](#) — Performs a hot deployment of a `Deployable` on a locally installed container
- [Remote Deployer](#) — Performs a hot deployment of a `Deployable` on a container running on a remote machine

## Example using the Java API

To instantiate a `Deployer` you need to know its class name. A `Deployer` is specific to a container (you can find the class names on the [container](#) page listing all containers).

The deployment is done using one of the `Deployer.deploy(...)` APIs. Some `deploy(...)` signatures accept a `DeployableMonitor` which is used to wait till the container has not finished deploying. Cargo currently offers a `URLDeployableMonitor` which waits by polling a provided URL (see below in the example). When the URL becomes available the monitor considers that the `Deployable` is fully deployed. In the future, Cargo will provide other `DeployableMonitor` such as a `Jsr88DeployableMonitor`.

## Example without using a DeployableMonitor

Hot-deploying a WAR on Resin 3.0.9 without waiting for the deployment to finish:

```
InstalledLocalContainer container = new
Resin3xInstalledLocalContainer(
    new
Resin3xStandaloneConfiguration("target/resin
3x"));
container.setHome("c:/apps/resin-3.0.9");

container.start();

DeployableFactory factory = new
DefaultDeployableFactory();
WAR war =
factory.createDeployable(container.getId(),
    "path/to/my.war",
    DeployableType.WAR);

Deployer deployer = new ResinDeployer();
deployer.deploy(war);
```

Please note that the `Deployer.deploy()` method call does not wait for the `Deployable` to be fully deployed before returning.

### Example using a `URLDeployableMonitor`

Hot-deploying an WAR on Resin 3.0.9 and waiting for the deployment to finish:

```
InstalledLocalContainer container = new
Resin3xInstalledLocalContainer(
    new
Resin3xStandaloneConfiguration("target/resin
3x"));
container.setHome("c:/apps/resin-3.0.9");

container.start();

DeployableFactory factory = new
DefaultDeployableFactory();
WAR war =
factory.createDeployable(container.getId(),
    "path/to/my.war",
    DeployableType.WAR);

Deployer deployer = new ResinDeployer();
deployer.deploy(war, new
URLDeployableMonitor("http://server:port/som
e/url"));
```

The <http://server:port/some/url> must point to a resource that is serviced by the Deployable being deployed.

### Example using the Ant tasks

Starting from CARGO version 1.1.0, the CARGO ANT tasks gained support for remote deployers. Here's a full example showing how to deploy a WAR to a remote Tomcat 6.x container.

```

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

<cargo containerId="tomcat6x"
action="deploy" type="remote">
  <configuration type="runtime">
    <property name="cargo.hostname"
value="production27"/>
    <property name="cargo.servlet.port"
value="8080"/>
    <property name="cargo.remote.username"
value="admin"/>
    <property name="cargo.remote.password"
value=""/>
    <deployable type="war"
file="path/to/simple-war.war">
      <property name="context"
value="application-context"/>
    </deployable>
  </configuration>
</cargo>

```

For more details, please check the example in the [Remote Container](#) section for the ANT tasks. The ANT tasks support the deployer actions `deploy`, `undeploy` and `redeploy`.

## Maven2 plugin

Please see the [Deploying to a running container](#) for a concrete example and the [Maven2 Plugin Reference Guide's <deployer> section](#) for full documentation.