Classpath resources in development mode

Classpath resource origin

Since version 5.4. Magnolia CMS introduces a concept of a `ResourceOrigin` - an abstraction which allows to develop different pipelines for resource loading in the system. One of them is the web application classpath - project developers can bundle their resources in the jars together with the java sources and Magnolia will recognise and expose such resources.

Here we will discuss how to deal with such resources in development mode when the application is deployed in debug mode from some IDE.

How do I find deployed classpath resources?

First of all - what do we imply under web application classpath? By default we mean the libraries residing in the **WEB-INF/lib** directory of the webapp. Note that not all the jar entries are considered to be resources by Magnolia (and all the jars considered to be valid resource locations). The default filters we apply can be deduced from the `DefaultClasspathServiceConfigurations` class (tl;dr - it excludes compiled class files, default and native java libraries, most of the Magnolia special directories which start with *mgnl-*, e.g. *mgnl-bootstrap*, **except** for *mgnl-i18n*).

In order to find out what is the exact version of a resource is deployed at the moment one has to know how their IDE sets up the webapp deployment, i.e. where is the deployed war file/directory. With Maven set-up it would typically use bundle project's **target** directory.

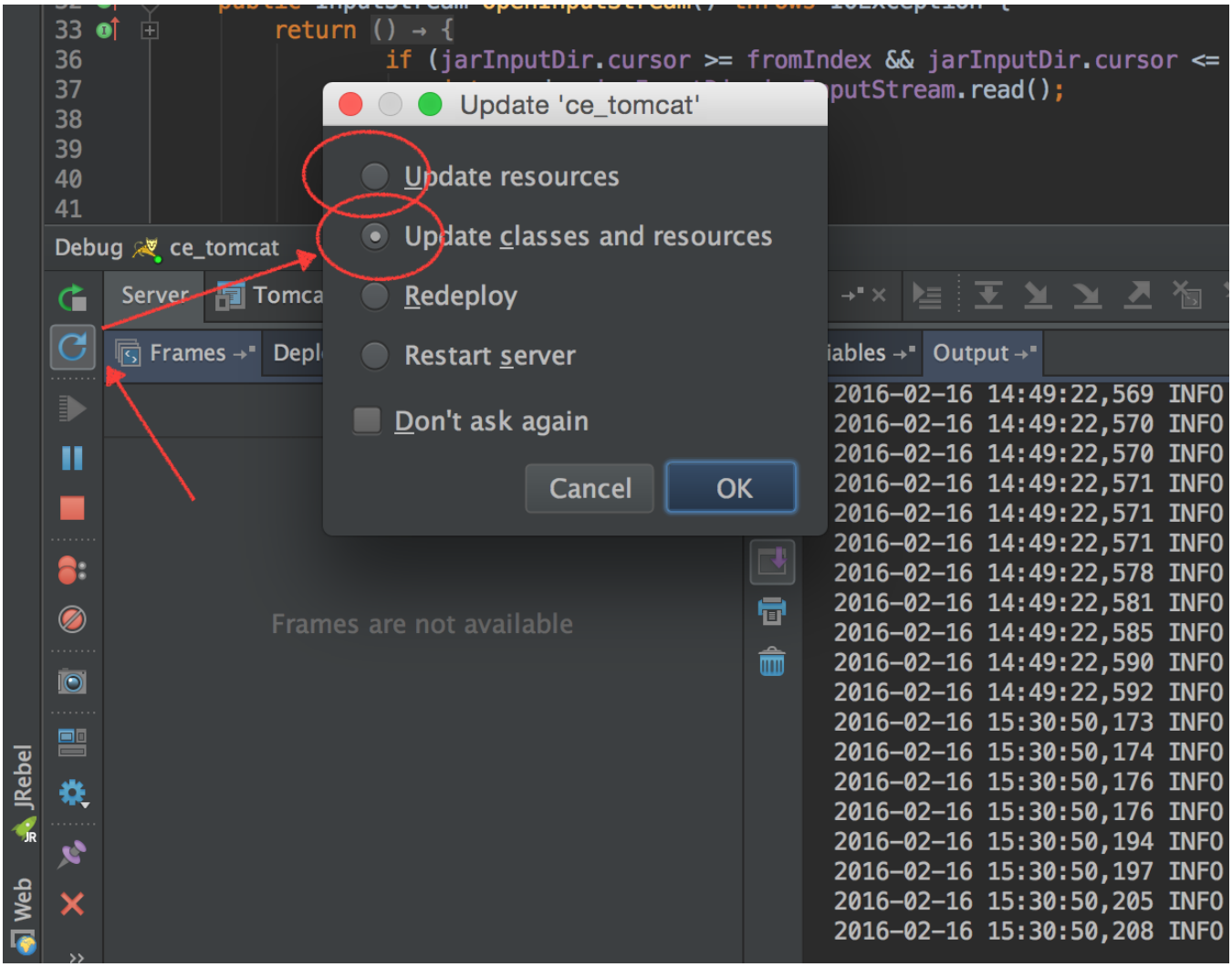
How do I re-deploy my resources?

First of all - the webapp has to be deployed in an exploded form (what's that?).

Once that is done - all you have to do is to get the latest resource version to the **WEB-INF/lib** folder and Magnolia will pick it up shortly (the monitoring interval is 10 seconds)! There are several ways to make such an update:

- One can manually put the new version of a jar with resources into **WEB-INF/lib**. This is, however, not the desired approach.
- Make your IDE refresh the resources. This is a generally better (and normally - easier) approach. However, just like with hot redeployment of java classes, this might be quirky.

IntelliJ



In IntelliJ IDEA given that the webapp is deployed as exploded war there should be a blue round arrow update button which brings a dialog box from which an *Update resources* or *Update classes and resources* should be triggered.

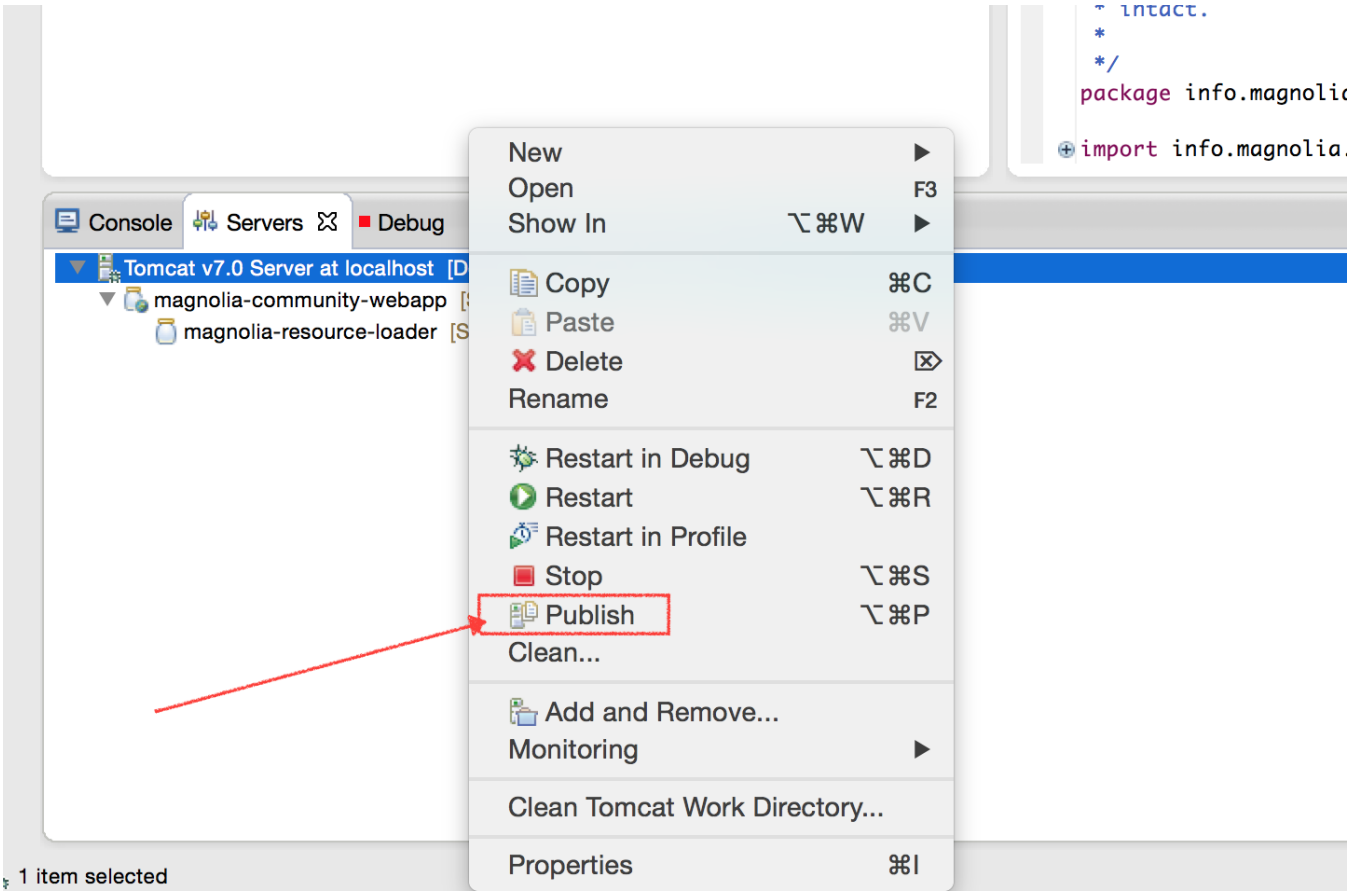
⚠️ **Quirks:**

It was noticed that IntelliJ is not always reliable when just the resources have changed (no java changes). Especially if a new resource has been added or some resource has been removed. One could try the following to avoid such situation:

- Make some trivial Java change and trigger an update in IDE - it was observed that often once is enough for the whole server uptime.
- When dealing with Maven - tests shown that single module projects are re-deployed much stabler than sub-modules of a multi-module ones.

Eclipse

In Eclipse case (provided that the webapp deployment follows the practices described here: Eclipse) the situation appears to be even better - the resource changes like modification/deletion/addition are automatically picked up swiftly in case automatic resource publishing is on. Should the automatic resource publishing be turned off - resources can published manually via server tab:



JRebel

Presence of JRebel should not affect the resource re-loading functionality. However, it doesn't seem to add any improvements in this sense either - one might still need to trigger resource update in IDE.

Mind that by default JRebel plugin seems to turn automatic resource publishing off!

Issues

Unfortunately there is a slight chance one might run into a JVM bug while re-loading classpath resources and experience a JVM crash. Some details can be found here:

http://stackoverflow.com/questions/1313071/jvm-crash-while-memcpy-during-class-load

The bug itself:

http://bugs.java.com/view_bug.do?bug_id=7129299