Application Configuration

This file is located in `WEB-INF` for a war and `META-INF` for an ear.

`@Asynchronous` configuration

Default pool size for `@Asynchronous` is 5. It can be very small for some applications highly relying on asynchronism or reactive patterns. Therefore it is possible to customize it adding these entries in `application.properties`:

Name	Default	Description
AsynchronousPool.Size	5	Core size of the pool
AsynchronousPool.CorePoolSize	5	Core size of the pool (inherit its default from .Size alias)
AsynchronousPool.MaximumPoolSize	5	Maximum size of the pool
AsynchronousPool.QueueSize	5	Maximum size of the pool
AsynchronousPool.KeepAliveTime	1 minute	Thread keep alive duration
AsynchronousPool.AllowCoreThreadTimeOut	true	Should thread timeout
AsynchronousPool.QueueType	LINKED (or SYNCHRONOUS if size == 0)	The type of queue of the pool in ARRAY, LINKED, PRIORITY or SYNCHRONOUS (same behavior as java implementations of the same name)
AsynchronousPool.ShutdownWaitDuration	1 minute	How many time to wait for the pool to shutdown when undeploying the application
AsynchronousPool.RejectedExecutionHandlerClass	-	A fully qualified name of a `java.util.concurrent.RejectedExecutionHandler`

TimerService and `@Scheduled`

`timerStore.class` allows to switch from the in memory (`org.apache.openejb.core.timer.MemoryTimerStore`) timer storage for quartz tasks to a custom implementation (using a database or anything for instance). Constructor can take a `TransactionManager` or nothing.

All quartz properties prefixed with `org.apache.openejb.quartz.` (instead of `org.quartz.`) are passthrough to quartz.

CDI

The boolean `openejb.cdi.skip-resource-validation` allows to not validate resources ie `@EJB` and `@Resource` usages in CDI beans.

All properties understood by OpenWebBeans will also be passthrough to OpenWebBeans from this location, see OWB config for more details.

@WebServiceRef

Name	Description
cxf.jaxws.client.wsFeatures	Allows to set WSFeature on the client injection. Values is a list (comma separated) of resource id in resources.xml or fully qualified names.

@Stateless

Name	Description
AccessTimeout or Timeout	container timeout
CloseTimeout	container timeout
BackgroundStartup	Don’t create instances in parallel if minimum count is > 0, default to false

resources.xml

`resources.xml` is a tomee.xml using application classloader.

As `tomee.xml` it supports filtering so you can use environment variables and system properties, for instance to use a MySQL database on OpenShift you can do:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<resources>
  <Resource id="MySQL" aliases="myAppDataSourceName" type="DataSource">
    JdbcDriver = com.mysql.jdbc.Driver
    JdbcUrl =
jdbc:mysql://${OPENSHIFT_MYSQL_DB_HOST}:${OPENSHIFT_MYSQL_DB_PORT}/rmannibucau?tcpKeep
Alive=true
    UserName = ${OPENSHIFT_MYSQL_DB_USERNAME}
    Password = ${OPENSHIFT_MYSQL_DB_PASSWORD}
    ValidationQuery = SELECT 1
    ValidationInterval = 30000
    NumTestsPerEvictionRun = 5
    TimeBetweenEvictionRuns = 30 seconds
    TestWhileIdle = true
    MaxActive = 200
  </Resource>
</resources>
```

`resources.xml` supports `Resource`, `Service` and `Container`.

`resources.xml` mecanism

`resources.xml` resources are still available globally like any `tomee.xml` resource.

The actual resource is bound in an application subtree called with the application name and a resource facade is bound in the global naming tree to be able to route the requests depending the application.

Typically if your application is named `myapp` and your resource id is `myresource` then instead of being registered as `myresource`, it will get registered as `myapp/myresource`.

If you get any ambiguity in resource name matching try to fully qualified your resource prefixing it with the application name.