# Transactional Mixin

## Overview

The **Transactional Mixin** is a mixin that allows you to add additional behaviour to the objects that you have defined as *transactional*.

It provides convience methods that the developer can use to interact with the Unit Of Work.

Here is the `org.codehaus.aware.unitofwork.Transactional` interface that you can use to access the introduced methods:

```
public interface Transactional {
    /**
     * Stores the persistable object in
persistent storage.
     */
    void create();

    /**
     * Removes the persistable object from
persistent storage.
     */
    void remove();

    /**
     * Clones the persistable object.
     *
     * @return a clone of the persistable
object
     */
    Object clone();

    /**
     * Checks if the persistable object
```

```
exists in the persistent storage.
     *
     * @return true if the object exists in
the persistent storage
     */
    boolean exists();

    /**
     * Marks the persistable object as
dirty. <p/>Is handled automatically by the
framework but this method exists for
     * the users to be able to do it
explicitly.
     */
    void markDirty();

    /**
     * Marks the current UnitOfWork and
attached transaction as restoreObject only.
     */
    void setRollbackOnly();

    /**
     * Returns the current UnitOfWork.
     *
     * @return the current UnitOfWork
     */
    UnitOfWork getUnitOfWork();

    /**
     * Returns the current transaction.
```

```
*
* @return the current transaction
```

```
     */
    TransactionContext getTransaction();
}
```

To access the methods you have to cast the instance to the interface:

```
((Transactional)this).setRollbackOnly()
```

or

```
((Transactional)targetInstance).create()
```

## Definition

The **Transactional Mixin** is an inner class of the UnitOfWorkProtocol and is defined using the `transactionalObjects` pointcut.

This is how you can define it in the XML definition:

```
<aspect
class="org.codehaus.aware.unitofwork.UnitOfWorkProtocol">
    <pointcut name="transactionalObjects"
expression="within(org.codehaus.aware.app.domain.*)"/>
    ...
</aspect>
```