

# The appearance of a well-tempered ICAT

## Contents

The appearance of a well-tempered ICAT .....	1
Table of figures .....	1
Introduction .....	1
The glassfish console.....	2
The jdbc connections .....	2
Ping the database .....	3
The deployed applications .....	4
The icat application.....	4
Using the icat tester to view the ICAT API .....	5
The ICAT Web Service tester.....	6
Invoke a method on the ICAT .....	7

## Table of figures

Figure 1: The Glassfish Console.....	2
Figure 2: The JDBC connections .....	2
Figure 3: Ping the database.....	3
Figure 4: The deployed applications .....	4
Figure 5: Icat application.....	4
Figure 6: Icat tester selector .....	5
Figure 7: Icat tester .....	6
Figure 8: getApiVersion.....	7

## Introduction

The purpose of this note is to provide notes for people attending the training on installing ICAT.

The note provides a series of screen shots with commentary. In order to keep the length of the document short, some intermittent steps are omitted.

The note describes the appearance of the ICAT when it has been installed correctly. The note does not describe how to configure it correctly.

It is probable that this note will be converted into a training document and added to the contrib directory of the icat distribution. I would appreciate comments, additions and corrections.

Alistair Mills  
October 2012

## The glassfish console

The glassfish console appears once the admin user has successfully logged on to the glassfish server. The welcome screen is shown in Figure 1: The Glassfish Console.

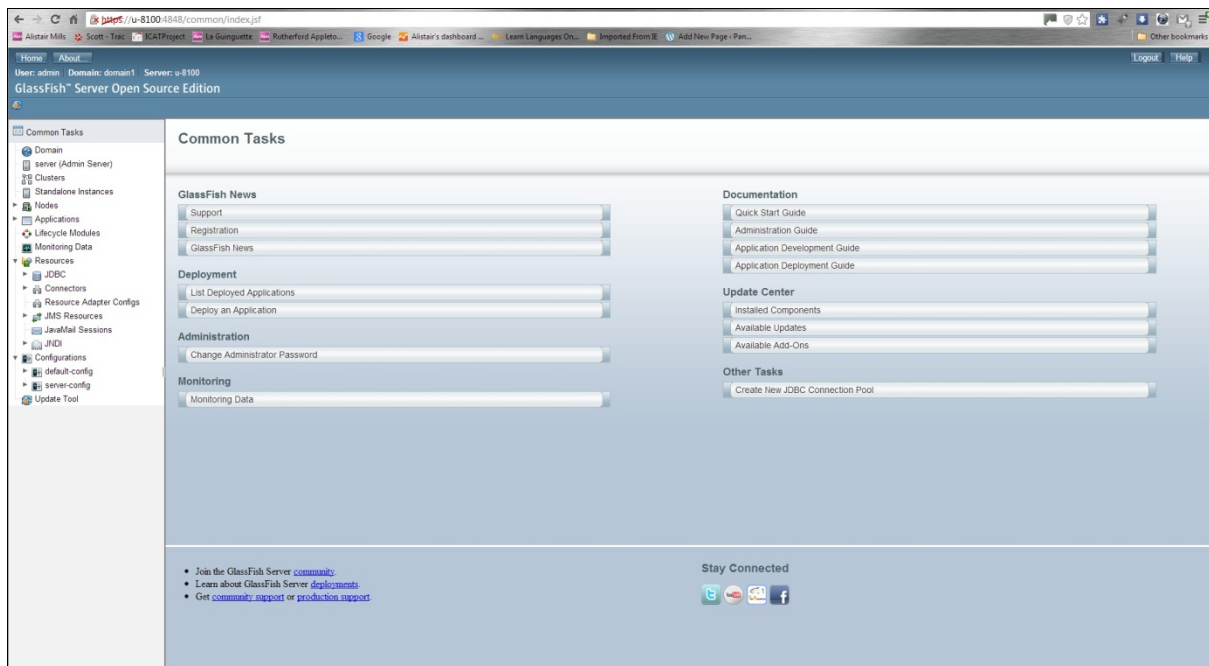


Figure 1: The Glassfish Console

## The jdbc connections

There are jdbc connections called authn\_db and icat. These connection pools identify the database to be used for authentication and for ICAT. Note that the icat does not know about the connections themselves. The icat expects that these connections are available for use. The jdbc connections are shown in Figure 2: The JDBC connections.

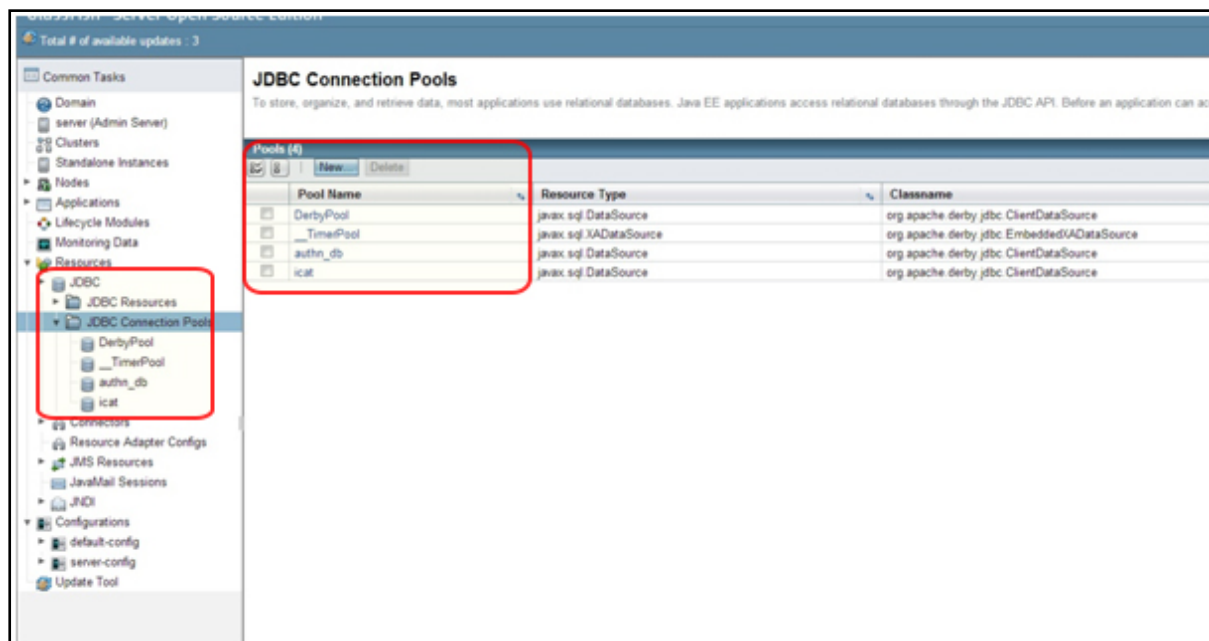


Figure 2: The JDBC connections

## Ping the database

The administrator can verify that the connections to the database are working. This is shown in Figure 3: Ping the database.

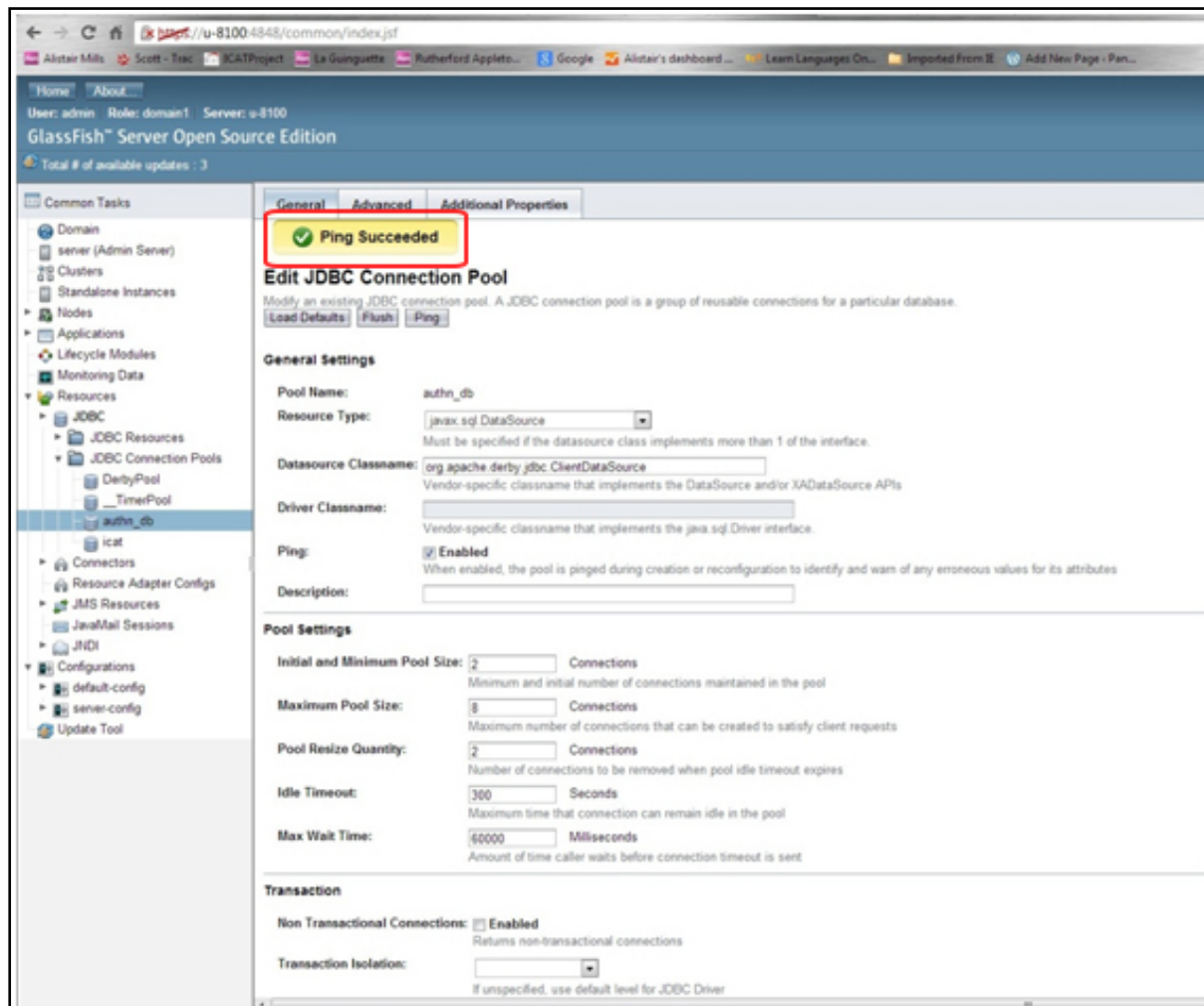


Figure 3: Ping the database

## The deployed applications

The administrator can view the state of the applications on the server. This is shown in Figure 4: The deployed applications.

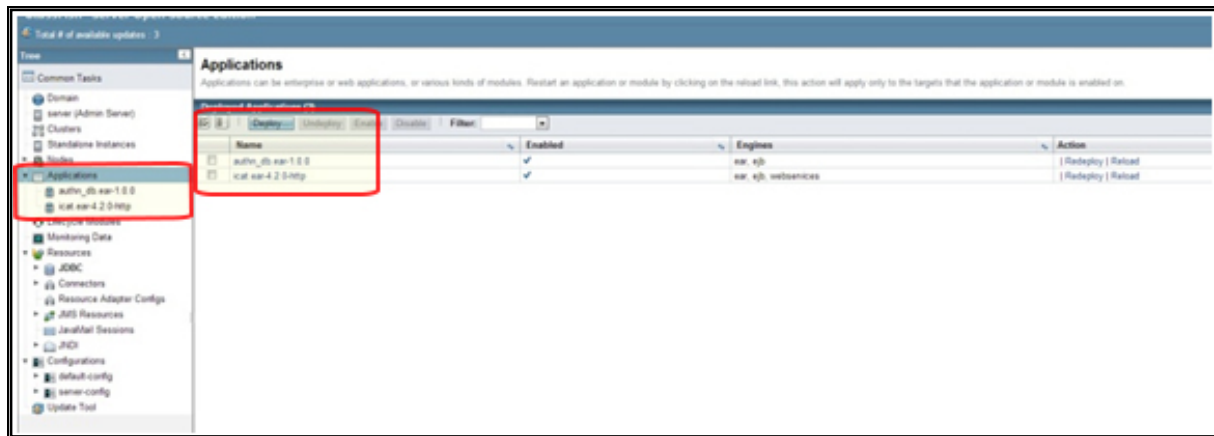


Figure 4: The deployed applications

## The icat application

The administrator can view the state of the icat application on the server. This is shown in Figure 5: Icat application.

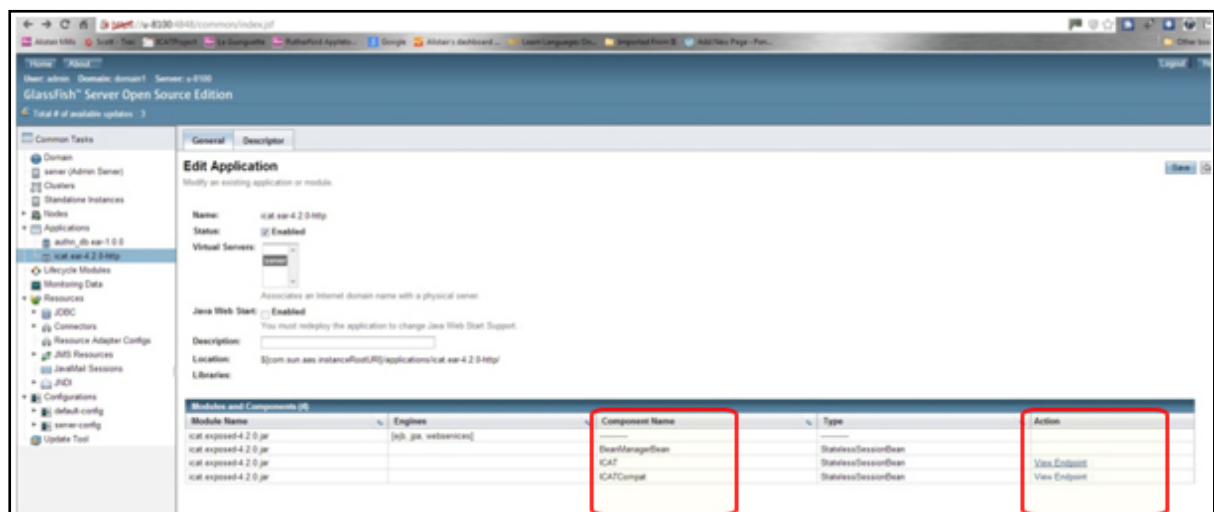


Figure 5: Icat application

## Using the icat tester to view the ICAT API

The view of the service end point is shown in Figure 6: Icat tester selector.

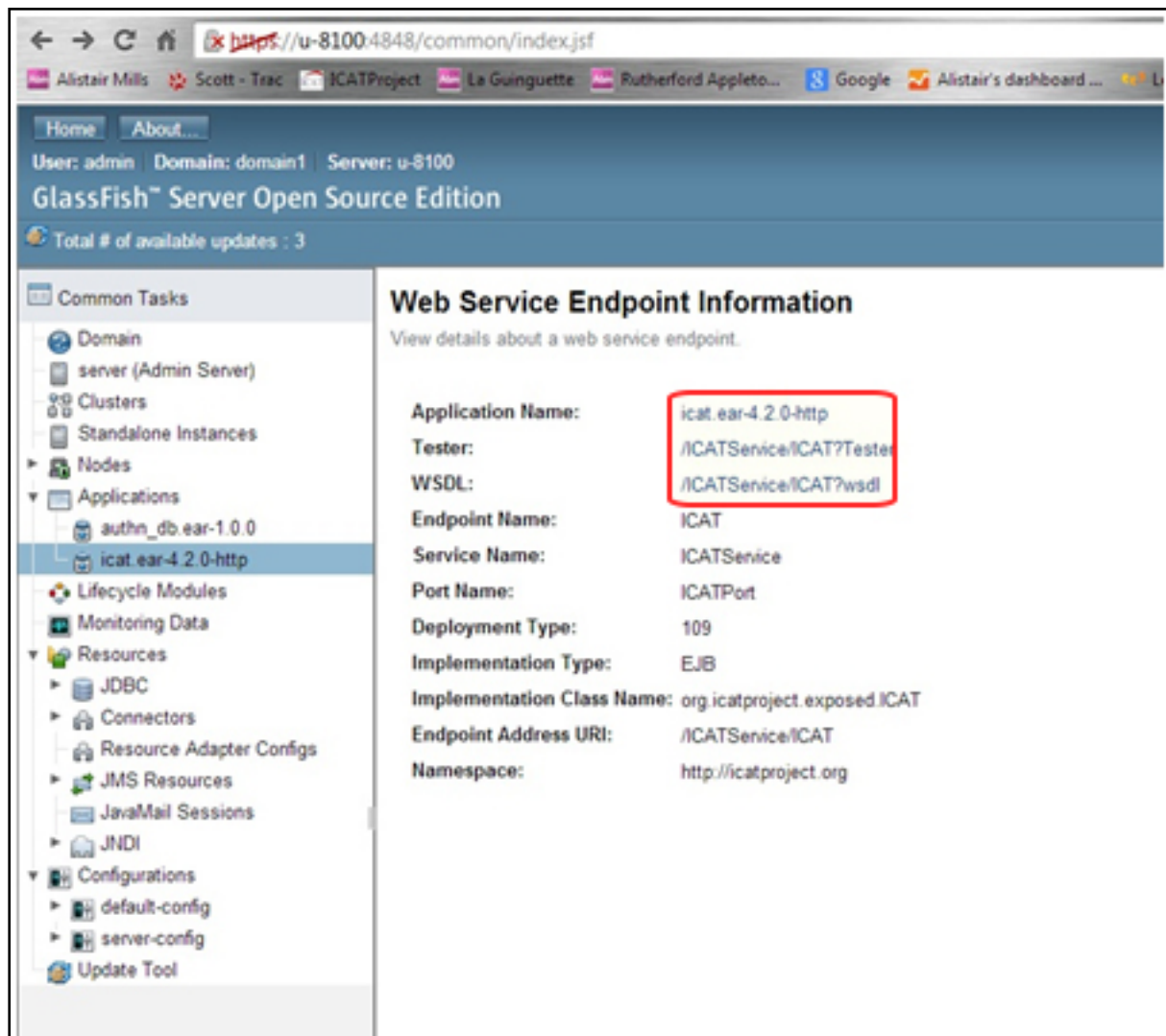


Figure 6: Icat tester selector

## The ICAT Web Service tester

When the icat tester is invoked, it displays a page as show in Figure 7: Icat tester.

ICATService Web Service Tester

This form will allow you to test your web service implementation ([WSDL File](#))

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

**Methods :**

public abstract org.icatproject.EntityBaseBean org.icatproject.ICAT get(java.lang.String java.lang.String java.lang.String) throws org.icatproject.IcatException\_Exception

public abstract void org.icatproject.ICAT delete(java.lang.String org.icatproject.EntityBaseBean) throws org.icatproject.IcatException\_Exception

public abstract java.util.List org.icatproject.ICAT search(java.lang.String java.lang.String) throws org.icatproject.IcatException\_Exception

public abstract void org.icatproject.ICAT update(java.lang.String org.icatproject.EntityBaseBean) throws org.icatproject.IcatException\_Exception

public abstract long org.icatproject.ICAT create(java.lang.String org.icatproject.EntityBaseBean) throws org.icatproject.IcatException\_Exception

public abstract void org.icatproject.ICAT dummy(org.icatproject.Datafile org.icatproject.DatafileFormat org.icatproject.DatafileParameter org.icatproject.Dataset org.icatproject.DatasetParameter)

public abstract java.lang.String org.icatproject.ICAT getUsername(java.lang.String) throws org.icatproject.IcatException\_Exception

public abstract java.lang.String org.icatproject.ICAT login(java.lang.String org.icatproject.LoginCredentials) throws org.icatproject.IcatException\_Exception

public abstract void org.icatproject.ICAT logout(java.lang.String) throws org.icatproject.IcatException\_Exception

Figure 7: Icat tester

## Invoke a method on the ICAT

The results from calling the getApiVersion method are shown in Figure 8: getApiVersion.

The screenshot shows a web browser window with the address bar displaying `u-8100:8080/ICATService/ICATTester`. The page title is **getApiVersion Method invocation**. Below the title, there is a section for **Method parameter(s)** with a table that has two columns: **Type** and **Value**. Below this, the **Method returned** section shows the result: `java.lang.String: "4.2.0"`, which is highlighted with a red rectangular box. Below the returned value, the **SOAP Request** section displays the XML payload used for the request. Finally, the **SOAP Response** section shows the XML payload received from the server.

**Method parameter(s)**

Type	Value
------	-------

**Method returned**

`java.lang.String: "4.2.0"`

**SOAP Request**

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:getApiVersion xmlns:ns2="http://icatproject.org"/>
  </S:Body>
</S:Envelope>
```

**SOAP Response**

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getApiVersionResponse xmlns:ns2="http://icatproject.org">
      <return>4.2.0</return>
    </ns2:getApiVersionResponse>
  </S:Body>
</S:Envelope>
```

Figure 8: getApiVersion