

# Beehive Sample: Petstore

## Table of contents

1 Introduction.....	2
1.1 Petstore Directory Structure.....	2
1.2 Build Targets.....	3
2 Running the PetStore Sample on Tomcat 5.....	4
2.1 To Set up the Environment.....	4
2.2 Download Derby Implementation.....	4
2.3 (Optional) Dowload HTTPUnit Implementation.....	4
2.4 To Copy the Petstore Application to a Project Folder (Optional Step).....	5
2.5 Edit the build.properties File.....	5
2.6 To Compile the Petstore Application.....	6
2.7 To Start Tomcat.....	6
2.8 To Deploy to Tomcat.....	6
2.9 Running Petstore.....	7
3 Running the Petstore Tests.....	7

## 1. Introduction

The Petstore sample demonstrates how to integrate all three Beehive technologies (NetUI, Controls and Web Services) in one application.

NetUI provides customers with web access to the Petstore. There are separate page flows for different customer activities: the 'shop' page flow (located at: `petstoreWeb/web/shop/`) lets users browse the catalog of pets, the 'checkout' page flow (`petstoreWeb/web/checkout`) lets users purchase selected items from a shopping cart, etc.

A web service (located at `petstoreWeb/ws/PestoreInventoryManager.java`) provides store managers access to the inventory.

Most of the page flows have associated Control files (located at: `petstoreWeb/src/org/apache/beehive/samples/petstore/controls/`). The Controls handle the backend data traffic and encapsulate the operations of the web application, such as retrieving data from a database and handling user orders.

Petstore runs against a Derby database implementation. See the instructions below for downloading Derby.

### 1.1. Petstore Directory Structure

The table below describes the functions of the most important files in the Petstore web application.

Directory	Description
<code>petstoreWeb</code>	
<code>src</code>	This directory contains the back-end data and controls used by the web app.
<code>org</code>	
<code>.../controller/AccountController.java</code>	Class controls some of the internationalization settings for the site.
<code>.../controls</code>	Beehive Controls encapsulating different functionalities, including handling user account data, catalogue data, and order processing.
<code>.../forms</code>	Form Beans: each form bean is a server-side representation of an HTML form appearing in the user interface.

.../model	Java object classes (java beans) used to represent the basic types in the app: addresses, orders, products, etc.
.../resources	Various .properties files
sql	A SQL file to initialize the database.
test-src	Java types used by the application's internal tests.
web	Contains the NetUI web application modules.
account	NetUI user interface for managing user accounts
auth	NetUI user interface for managing login
checkout	NetUI user interface for managing the shopping cart
search	NetUI user interface for searching the Petstore
shop	NetUI user interface for browsing the catalogue of pets
site	Contains the Petstore "skin", including the CSS file, HTML frameset, and template file
webappRoot	
WEB-INF	Contains configuration files for deployment, validation, security (web.xml), and generated Struts config files.
tags	Contains .tag files.
lib	JAR resources -- this directory is created and populated when you call the ant target deploy-beehive.
ws	Contains the web service interface for the web app.

## 1.2. Build Targets

The build file (located at `petstoreWeb/build.xml`) contains the following build targets.

Target	Description
--------	-------------

deploy-beehive	Copies the Beehive runtime JARs into the webapp's WEB-INF/lib directory.
build	Compiles the webapp components: XMLBean schemas, controls, page flows, and web services.
scrub	Deletes the build directories and the test-related directories.
clean	Deletes the build directories from the web app.
war	Builds a WAR file from the web app files.

## 2. Running the PetStore Sample on Tomcat 5

The following explains how to run the Petstore sample on Tomcat 5. The Petstore sample will run on other web containers, but we have chosen Tomcat 5 for convenience.

### 2.1. To Set up the Environment

Before proceeding, complete all of the necessary and optional steps in the following topic: [Beehive Installation and Setup](#)

Open a command shell and confirm that you have set following variables:

- ANT\_HOME
- JAVA\_HOME
- CATALINA\_HOME

Also ensure that the following elements are on your PATH:

- ANT\_HOME/bin
- JAVA\_HOME/bin

### 2.2. Download Derby Implementation

The Petstore app runs against an imbedded Derby database. You must download a Derby implementation before building and running the Petstore.

Download a Derby distribution at: [http://db.apache.org/derby/derby\\_downloads.html](http://db.apache.org/derby/derby_downloads.html)

### 2.3. (Optional) Dowload HTTPUnit Implementation

If you want to run the HTTPUnit tests associated with Petstore, you must download a

HTTPUnit implementation from: <http://httpunit.sourceforge.net/>

## 2.4. To Copy the Petstore Application to a Project Folder (Optional Step)

To keep your Beehive distribution directory pristine, you should copy the folder `<BeehiveRoot>/samples/petstoreWeb` to another location before proceeding.

### Note:

`<BeehiveRoot>` refers to the top-level directory of your Beehive installation. A typical value for `<BeehiveRoot>` would be `/apache/apache-beehive-1.0.`

The following instructions assume that you have copied the folder `petstoreWeb` into the directory `/beehive_projects`, resulting in the following directory structure.

```
/
beehive_projects
  petstoreWeb
```

### Note:

Strictly speaking, you do not need to copy the `petstoreWeb` directory to another location. If you wish to leave the `petstoreWeb` directory in place, in the instructions below you must replace occurrences of the path element `/beehive_projects` with this path element: `<BeehiveRoot>/samples`.

For example, to build the sample, run the following Ant command:

```
ant -f <BeehiveRoot>/samples/petstoreWeb/build.xml clean build war
```

To delete the sample's build directory, run the `clean` target:

```
ant -f <BeehiveRoot>/samples/petstoreWeb/build.xml clean
```

## 2.5. Edit the build.properties File

Edit the `build.properties` file--the file that sets the build-related properties for your web application.

Open the file `/beehive_projects/petstoreWeb/build.properties` in a text editor.

At a minimum, you will need to edit the following two properties:

- `beehive.home`
- `derby.jar`

If you want to run the `httpunit` tests, you must set the following additional property

- `httpunit.dir`

The following example shows typical values for these three properties. Note that the property `derby.jar` points to a JAR file, not to a directory.

```

beehive.home=/apache/apache-beehive-1.0

context.path=petstoreWeb

servlet-api.jar=${os.CATALINA_HOME}/common/lib/servlet-api.jar
jsp-api.jar=${os.CATALINA_HOME}/common/lib/jsp-api.jar

derby.jar=/apache/derby/derby.jar

# Set this path in order to run HTTPUnit tests against the webapp
httpunit.dir=/httpunit-1.6

```

**Note:**

Windows users must use forwardslashes (/) not backslashes (\) in the build.properties file.

## 2.6. To Compile the Petstore Application

To compile the Petstore app, enter the following Ant command:

```
ant -f /beehive_projects/petstoreWeb/build.xml clean build war
```

A WAR file named petstoreWeb.war will be created at  
/beehive\_projects/petstoreWeb/petstoreWeb.war.

## 2.7. To Start Tomcat

To start Tomcat, run the following command:

```
%CATALINA_HOME%\bin\startup.bat
```

On non-Windows systems:

```
$CATALINA_HOME/bin/startup.sh
```

## 2.8. To Deploy to Tomcat

To deploy the web application, copy the WAR file to Tomcat's webapps directory.

On Windows:

```
copy C:\beehive_projects\petstoreWeb\petstoreWeb.war
%CATALINA_HOME%\webapps /Y
```

**Note:**

On Windows, there are file-locking issues that Tomcat versions 5.5.x and above are sensitive to. In particular, any web application that uses Struts will **fail to redeploy** if you copy in a new .war file as described here. The Commons Digester team is adding a workaround for the issue (see [this bug](#)), but in the meantime, you can work around it with the antiResourceLocking option in Tomcat. Just add a file called context.xml in a directory called META-INF inside the web directory before building (so it will end up as META-INF/context.xml in your controls\_tutorial.war):

```

<?xml version="1.0" encoding="UTF-8"?>
<Context antiResourceLocking="true">
</Context>

```

Everywhere else:

```
cp /beehive_projects/petstoreWeb/petstoreWeb.war $CATALINA_HOME/webapps
```

## 2.9. Running Petstore

Visit the following URL:

<http://localhost:8080/petstoreWeb>

## 3. Running the Petstore Tests

The Petstore app ships with associated junit tests. Before running the tests, ensure that the web app is up and running, i.e, accessible at <http://localhost:8080/petstoreWeb>.

To run the tests, enter the following Ant command:

```
ant -f /beehive_projects/petstoreWeb/build.xml test.run
```

Java, J2EE, and JCP are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

© 2004, Apache Software Foundation