Using the wizards and tools

Table of Contents

Chapter 1. Introduction

Kelp provides wizards and tools that help you develop Enhydra applications from within your IDE.

Chapter 2. Using the Kelp Application Wizard

The Kelp Application Wizard, shown in Figure 1, speeds up project development by generating the framework and source files for new applications and components. The Kelp Application Wizard uses two generators to create Enhydra projects: the Web Application generator and the Enhydra Application generator.

The Application Wizard uses the following generators:

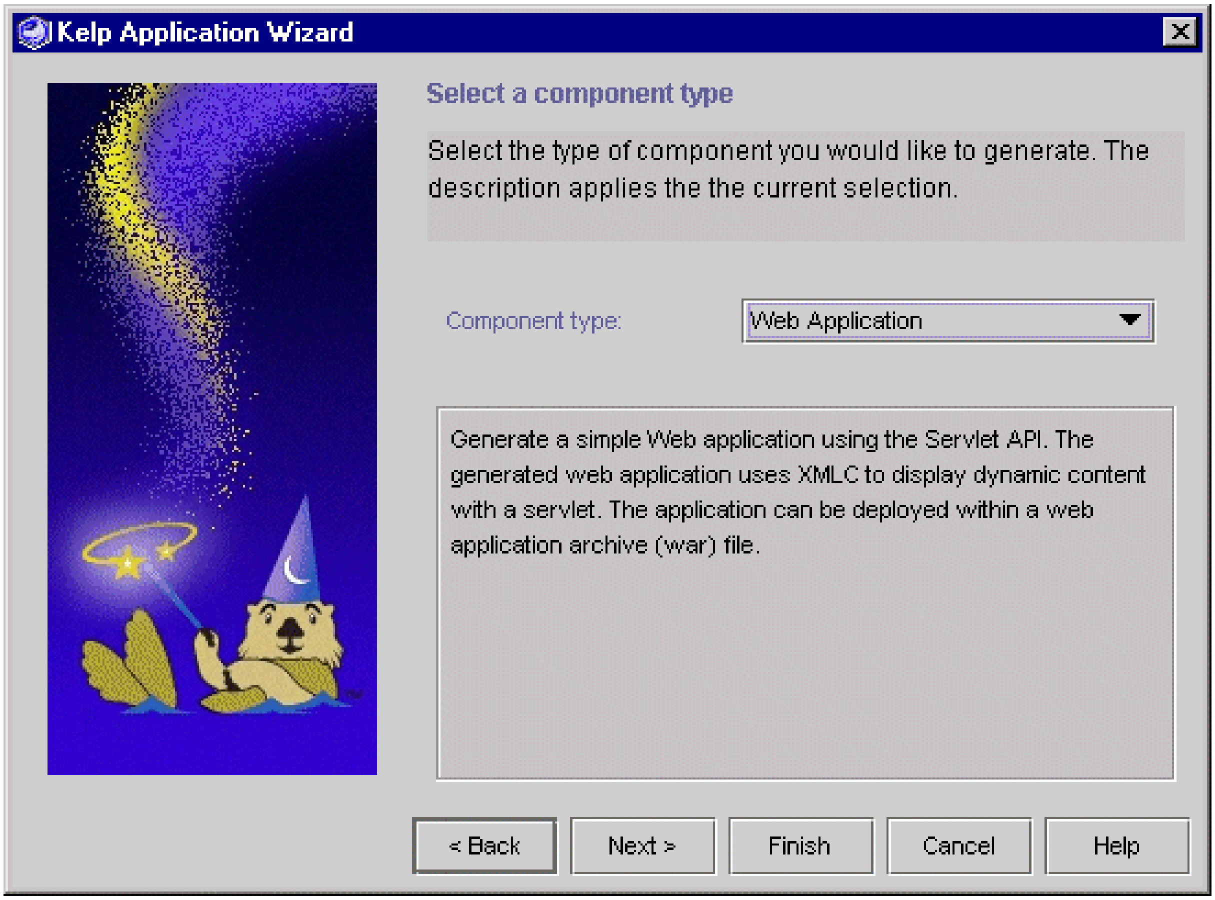
- Web application (Servlet 2.2 compatible)
- Enhydra application (super-servlet style application)

Creating a new project:

To use the Kelp Application Wizard from within the IDE, create a project beforehand. The NetBeans Project must have a mounted directory into which you will generate your project files.

Important: Running the Kelp Application Wizard while you are using a project that already contains files generated by the Kelp Application Wizard might cause unexpected results. Before you run the Kelp Application Wizard, create a new project for the generated files.

- To generate project file in NetBeans, select File|Kelp Application Wizard to launch the Kelp Application Wizard. Then, choose the component type to generate from the Component Type pull-down menu.



Select a component type

Select the type of component you would like to generate. The description applies the the current selection.

Component type: Web Application

Generate a simple Web application using the Servlet API. The generated web application uses XMLC to display dynamic content with a servlet. The application can be deployed within a web application archive (war) file.

Figure 1: Kelp Application Wizard, launched from NetBeans

Note: To create a new project in JBuilder, select File|New to open the Object Gallery as shown in Figure 2 . Click the Enhydra tab, select the component type to generate, and click OK. The Object Gallery in JBuilder replaces the first page of the Kelp Application Wizard. The rest of the wizard is unchanged.

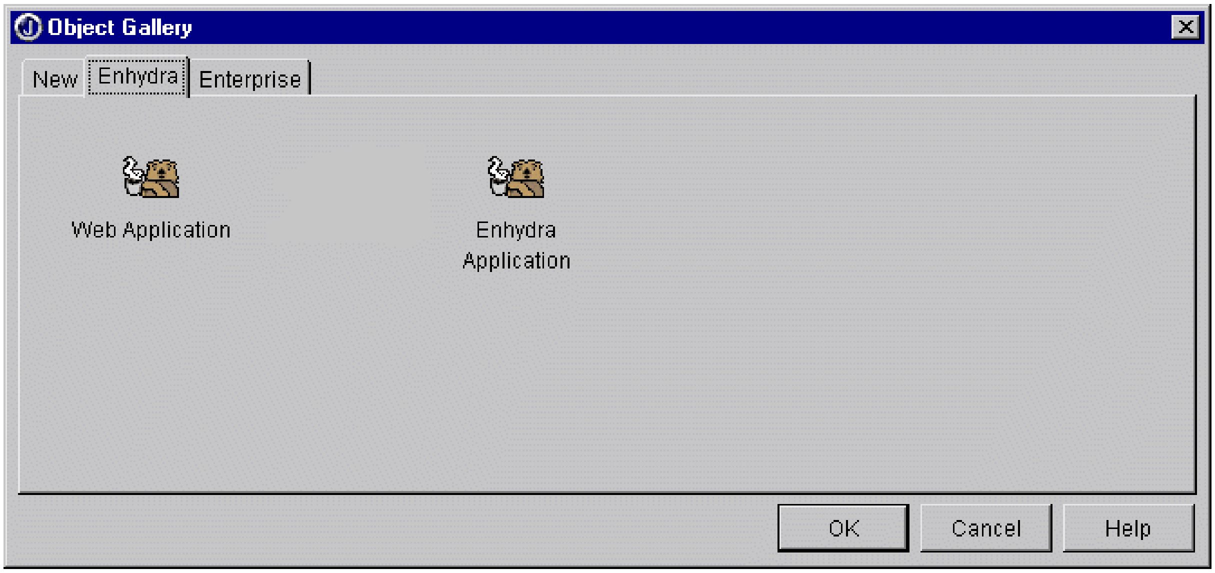


Figure 2: Object Gallery in JBuilder, showing choices for generating Enhydra compontents

- Navigate through the wizard and select the options and naming conventions for your genenerated files.
- When you have finished setting the options you want, click Finish to generate your application or service. When generation is complete, view the newly generated Readme.html file that has been added to your project. Readme.html shows the steps you need to build and run the application or service. The steps will vary, depending on which IDE you are using.

For more information on the Kelp Application Wizard, see Chapter 2, "Using the Kelp Application Wizard," of the Developer's Guide. Once you create a project, you can use the Kelp XMLC tool and the Kelp Deployer tool to work with it.

In both the Kelp XMLC tool and Enhydra Deployment wizards, you use a tabbed dialog box to select files, set options, and view the build process.

Chapter 3. Enhydra Import Wizard

The Enhydra Import wizard is similar to the Application Wizard. This wizard allows you to convert a GNU Makefile project for an Enhydra Application to an IDE project.

Note: The Enhydra Import Wizard is only supported by JBuilder. For additional information, refer to "Using the Enhydra Import Wizard" (html [../using_kelp.html#ch05] , pdf [../using_kelp.pdf]), in the fifth chapter of the document "Using Kelp with an IDE" (html [../using_kelp.html] , pdf [../using_kelp.pdf]).

Chapter 4. Using the Kelp XMLC tool

The XML Compiler dialog calls XMLC, which compiles HTML and XML files into DOM classes. To open the dialog, select Tools|XML Compiler. The tool is available only when a project is open.

The dialog box has three primary tabs: Selections, Options, and Output. The Selections tab, shown in Figure 3, displays all files in the currently selected project that are recognized as compilable by XMLC. You can use the single arrow buttons (< and >) to add or remove files from the list selected to be compiled. Use the double arrow buttons (< < and > >) to add or remove all files from the selection list. Select the Show Full File Path check box to display the full path of the files.

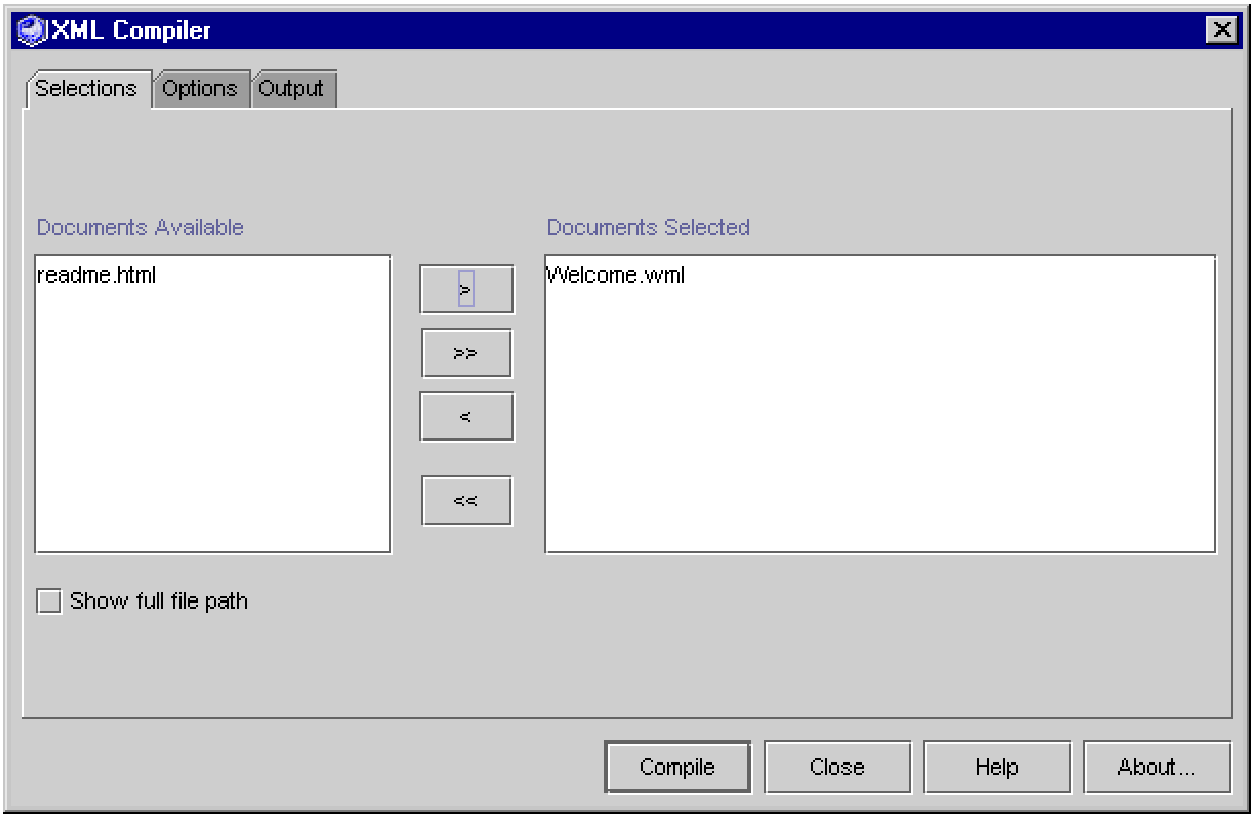


Figure 3: Selections tab of the XML Compiler dialog

The Options tab displays additional tabs for compile, XMLC types, and trace options.

- The Compile options let you customize the generated DOM classes.
- The XMLC Types tab allows you to add new file extensions to associate types of files with XMLC. Files with the specified extentions display on the Selections tab and can be selected if you want them to be processed by XMLC.
- The Trace options let you display detailed information during the compile process without affecting the generated classes.

Note: In JBuilder, the Options tab also has an Invoke XMLC during Project Make/Rebuild check box. Select this check box to call XMLC without opening the XML Compiler dialog. This calls XMLC when you build or make a project node that contains a selected XMLC document type file.

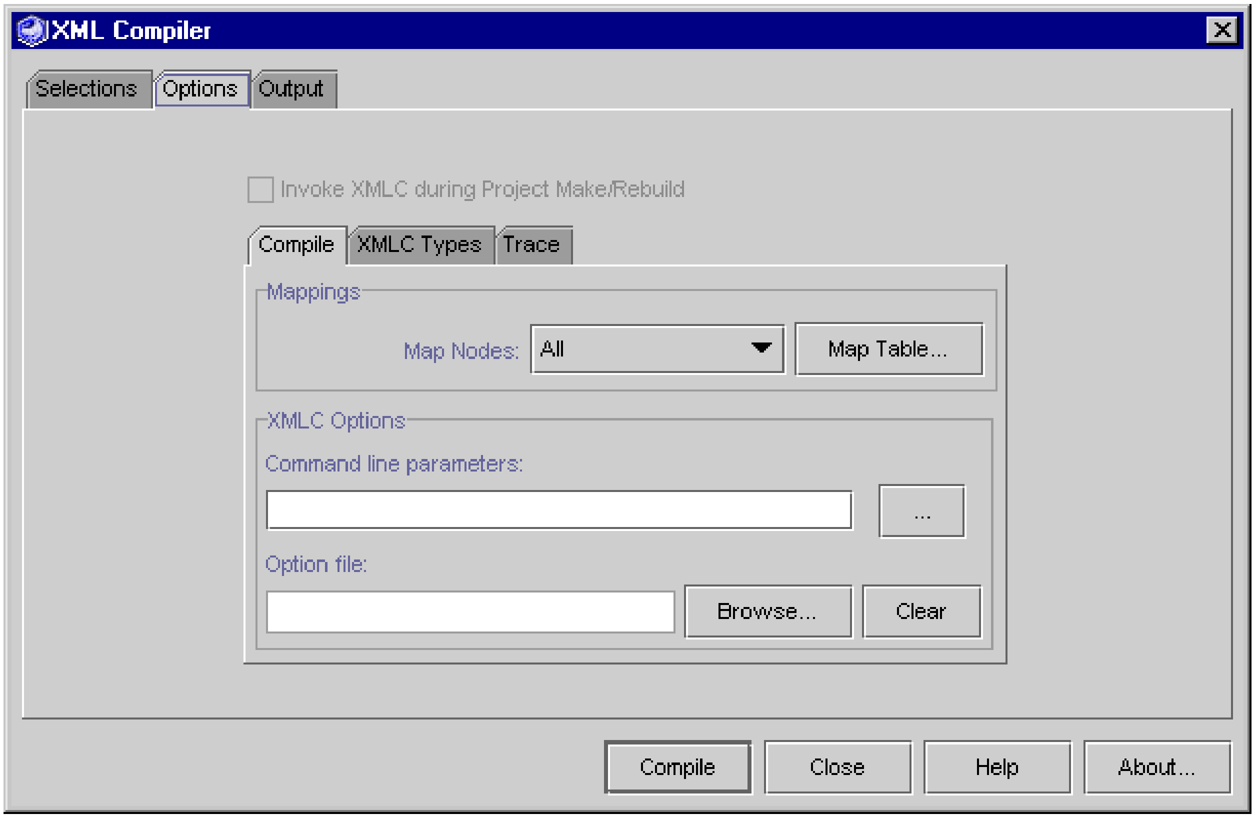


Figure 4: Options tab of the Kelp XMLC tool

The Output tab shows the results of running XMLC based on the files you selected on the Selections tab. For information on the Output tab, see "Setting output options".

Using mapping tables for generated class names

One common option is to use a mapping table to customize generated class names. By default the XML Compiler dialog uses the current directory name to determine the package name in the generated source. Unless your HTML files are stored in the same directory as your presentation package Java source files, you need to use the mappings option to map the resource document directory to the presentation package name.

Note: The Enhydra guidelines recommend that you keep your XMLC documents in a resources directory and map them to the presentation package when you are generating the DOM classes. Both the Kelp sample project and the DiscRack example store HTML files in a resources directory.

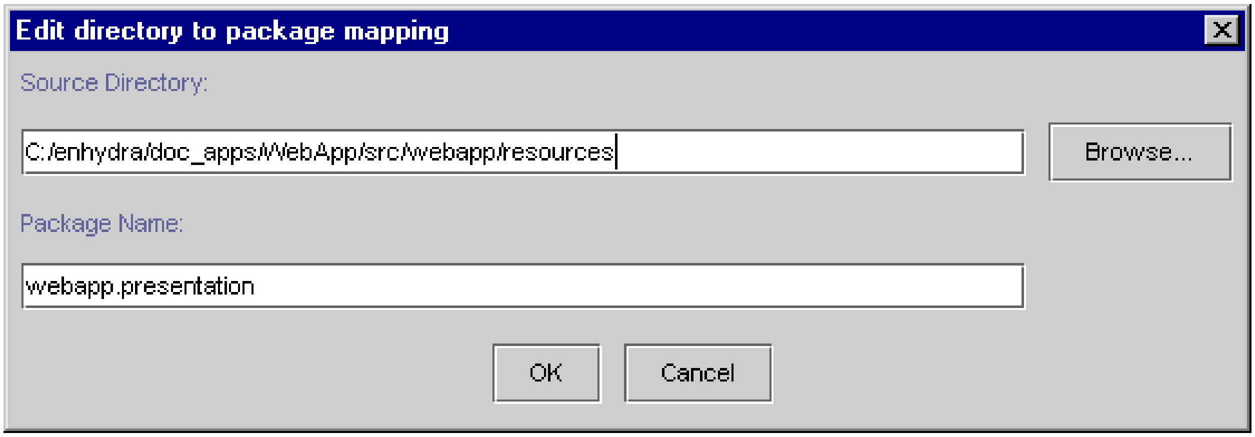
To create mappings:

- Click Edit in the Mappings section of the Make tab. This opens the Project Map editor, which lets you associate source directories with package names.
- Click Add to create a new mapping.

  You can enter a source directory or use the Set button to navigate to one.

The dialog box in Figure 5 shows how to set up the compiler to use the kelp.webapp.presentation package name when you are compiling HTML pages stored in

`D:/jbuilder5/kelp5/samples/webapp/src/kelp/webapp/resources.`



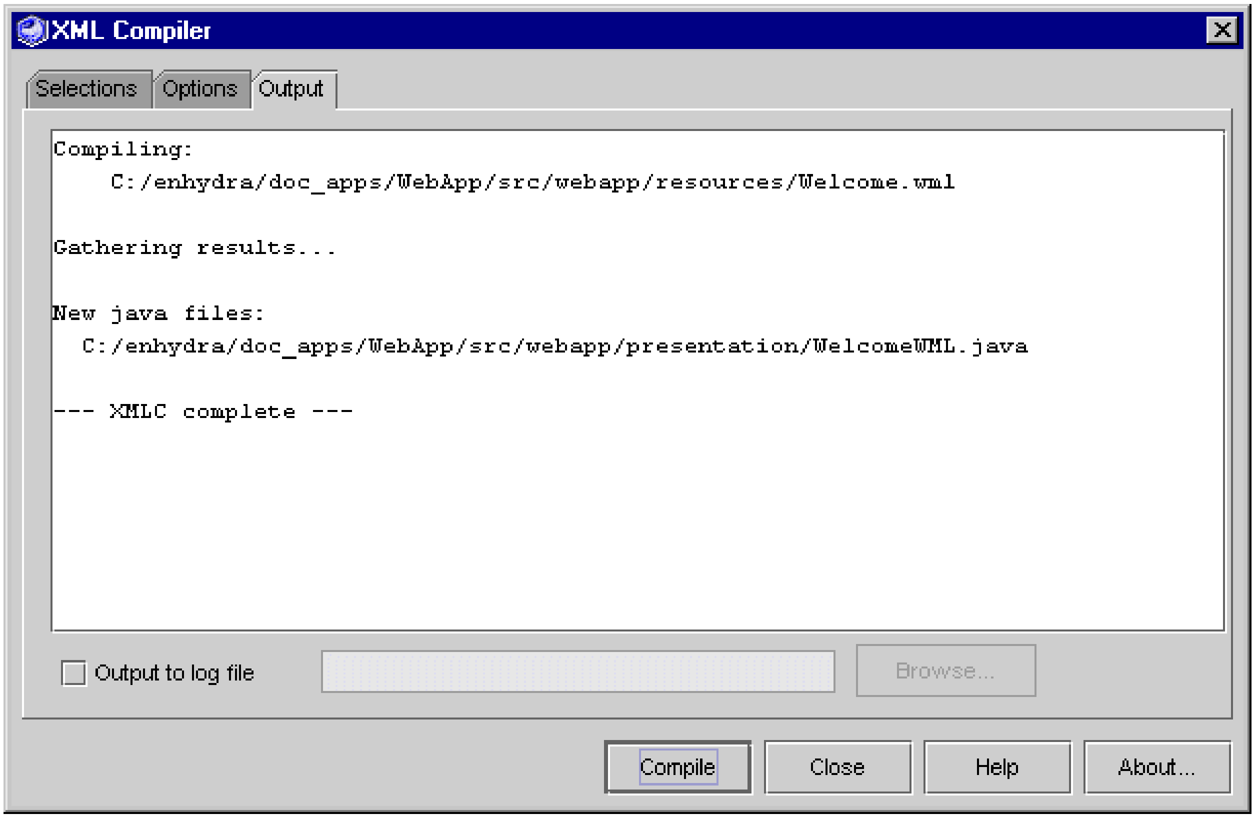
Source Directory: C:/enhydra/doc_apps/WebApp/src/webapp/resources

Package Name: webapp.presentation

Figure 5: Mapping a source directory to a package

Setting output options

The Output tab (Figure 6) is automatically selected when you click Compile in the XML Compiler dialog. This tab contains a scrollable text area that displays the results of the compile. If you have any errors in your HTML files, the problems appear on this tab. You can optionally save the output to a text file by selecting Output To Log File and entering a file name.



```
Compiling:
    C:/enhydra/doc_apps/WebApp/src/webapp/resources/Welcome.wml

Gathering results...

New java files:
  C:/enhydra/doc_apps/WebApp/src/webapp/presentation/WelcomeWML.java

--- XMLC complete ---
```

Figure 6: Output tab of the XML Compiler dialog

The Output page displays the files that are created during the compilation process. At the start of the compilation process, the dialog erases any files it created during a previous compile. If an HTML file contains a new error, the associated Java and class files are erased and not regenerated.

Chapter 5. Using the Kelp Deployer

The Kelp Deployer lets you quickly configure projects for your current environment and directory structure. The Kelp Deployer helps you perform the following four tasks:

- Generate configuration files and deployment descriptors from input templates
- Copy static content to your document root
- Create a deployable archive, based on the type of application or component
- Set up your project so you can launch Enhydra with your application

General Tab

This tab sets and displays basic information about deployment. The messaging options control how status messages are handled during deployment.

- Deploy Root--Deploy Root is used for setting the archive document root directory. Click the Ellipses (...) button to navigate to a folder.
- Deploy Type--Select the Enhydra component to deploy from this pull down menu.
  - Web Application--If your project uses the Java servlet API, select Web Application. Servlet archives are Web Application Archive (WAR) files that contain the output classes, static content files, and a deployment descriptor.
  - Enhydra Application--If you are using the Enhydra super-servlet API, select Enhydra Application. Enhydra super-servlet archives contain the classes in your output directory.
- Deploy Messages
  - Messages--Shows status messages during deployment.
  - Options

    Display During Project Make/Rebuild--Used in JBuilder when the Kelp Deployer is set to invoke during project building.

    Write to File--Allows you to save all messages to a log file.
- Deploy During Project Make/Rebuild check box--Select Deploy During Build to deploy your application on each project rebuild. The Show Messages box is enabled if Check Deploy During Build is selected. If Show Messages is selected, deployment messages will be shown in the Output tab.
- Overwrite Without Warning--Normally, you want this checked. Otherwise, a warning dialog box is opened for every file that is redeployed.

Input Tab

Template Path points to the location of your project's .in files, which are template files. These files are processed and copied to your Deploy Root folder using relative paths.

- Selections--The Selections tab lets you select which of the available template files in your project will be processed. The left pane shows available templates, and the right pane shows the selected templates. Use the arrow buttons to move files from one pane to the other. Double arrow buttons (<< and >>) move the entire contents of one pane to the other. Single arrow buttons (< and >) move the selected file or files from one pane to the other.

  Select Show Full File Path to show the full path of each template file.

- Replacements--The Replacements tab lets you adjust the template settings for your project. Templates have the extension .in. Templates are text files with placeholders that are replaced with system-specific information by the Kelp Deployer. One of the default placeholders, @CLASSES@, is replaced with the current class output directory specified by your project.

  You can customize the search and replace mechanism by editing the data in the Replace Text table on the Options tab of the dialog. The default option lets you quickly restore the default project settings. The Replace With values can refer to relative paths. If you enter a value starting with a period, the Kelp Deployer substitutes the directory containing the project file for the period. For example, if you open the Kelp Web Application project from:

  ```
  /home/user/jbprojects/samples/webapp/KelpWebApp.jpr
  ```

  and have a Replace Text table containing Text to Find: @CLASSES@ Replace With: ./classes

  The resulting configuration files (.conf) will contain /home/user/jbprojects/samples/webapp/classes in place of the @CLASSES@ placeholders that are in each of the templates (.conf.in).

  For applications deployed on Windows, some Text to Find values of the form @*_PATH@ are handled differently than typical Text to Find values. They are expanded according to the following find values to deal with the different forms of Windows paths:

  - @SHELL_*_PATH@ expands to UNIX-style paths with the system drive as the root of the file system and the //<drive_letter>/<dir> format for other drives. For example, //d/myDir
  - @JAVA_*_PATH@ expands to Java-style Windows paths with the <drive_letter>:/<dir> format. For example, d:/myDir.
  - @OS_*_PATH@ expands to Windows-style paths with the <drive_letter>:\<dir> format. For example, d:\myDir.

  You can change the order for replacements using the Move Up and Move Down buttons. Items at the top are replaced first. The Reset button sets the Replace With text back to the default values.

- Show Full File Path--Displays the full path for files

- Input Templates (.in) Only--If not checked, you can have the Kelp Deployer copy (without searching and replacing strings) all non-template (.in) files from the input root directory to the deploy root directory.

- Enable Input Deployment--Enables or disables input deployment. If you are using auto-deploy, you can normally disable input deployment.

Content Tab

The Kelp deployer will copy content types from the content source directory to the archve document root. Use the archive document root to select content when building Web application archives.

- Paths--If you Select Enable Content Deployment, you can select the directory containing the static content files you want to deploy with your Web application.

  - Content Source--The directory that contains the static content files for your project. These static files will be copied to your document root using relative paths.
  - Archve Document Root--This field cannot be edited. It lists the directory where the archive file will be created.

- Types--The Content Types subtab shows the extensions of static resource files that will be copied from the Content Source directory to your document root directory. Click Add to enter a new extension. Select an extension and click Remove to keep files with that extension from being copied. Click Reset to go to the default values of recognized extensions.

Archive Tab

The Kelp Deployer can create and recreate archive files that you can auto-deploy to a running Enhydra server. The table lists the archives that have been or will be generated during deployment. If the Built check box is selected for an item, then a deployable archive already exists.

Clicking Edit opens the Kelp Archive Wizard. The wizard lets you create an archive. For more information about the Kelp Archive Wizard, refer to the Chapter 3, "Using the Kelp Archive Wizard," of the Developer's Guide.

Run Tab

The deployer can deploy archives to a Enhydra server running locally or set up your project so you can launch it from the IDE.

- Auto-Deploy to Locally Running Server--
- Confgure Project for Starting Server from IDE--

Chapter 6. Deployment example: deploying a Web application

After you have generated and built a Web application, you can deploy the Web application archive (WAR) in a few different ways using the Kelp Deployer.

To run the application manually on a restricted instance of the Enhydra server:

Kelp can generate scripts for running your application locally in an isolated instance of the Enhydra server. This instance of the Enhydra server loads the minimum number of services required to run your application. This configuration of the Enhydra server does not support the use of the LMC or the Web console.

- Select the Run tab in the Kelp Deployer dialog.
- Select Configure Project For Starting Server >From IDE.
- Click the Deploy button.
- In a shell window, change directories to <project_root>/output.
- Execute the start script, run or run.bat, to launch a local instance of the Enhydra server.

  This instance of the Enhydra server loads only the services required to run your application. The application is started automatically.

  Note: You may need to set the script to be executable.
- When Enhydra server is finished booting, access the application with a browser at the following URL:

  `http://localhost:8002`

  You should see the Welcome.html page with a redirect and a dynamic time stamp.

To hot deploy your application:

Hot deployment consists of deploying an application or component to a running instance of Enhydra. The Kelp Deployer can only deploy to a Enhydra server running locally (i.e., running on a mapped drive).

- Execute the Enhydra server start script, multiserver or multiserver.bat in <Enhydra5.1_root>/bin to start the server

  note: It is important to launch the Enhydra server from the /bin directory.
- Select the Run tab in the Kelp Deployer dialog.
- Select Auto-deploy to Locally Running Server.
- Click the Deploy button.

  The archive file is copied to <Enhydra5.1_root>/deploy directory. The Enhydra server will deploy the file automatically. When the file is deployed by Enhydra, the WAR file is removed from the /deploy directory.

- Start the Enhydra Multiserver Admin Console.
- To start your application, log in to the Enhydra server to which you deployed your application.

  The default username and password are "admin" and "enhydra" respectively.
- In the LMC Browser, select the Web application, and click Start in the Operations section displayed in the Workspace.
- Access the application with a browser at the following URL:

  `http://localhost:8002/appname`

  Where appname is the name of your WAR. The name of the WAR is used as the URL prefix by default. You should see the Welcome.html page with a redirect and a dynamic time stamp.

To run your application from within the IDE:

- Select the Run tab in the Kelp Deployer dialog.
- Select Configure Project For Starting Server >From IDE.
- Select Create StartServer.java
- Click the Deploy button.
- Find the generated StartServer.java file in the project source directory.
- Right-click the node and select Compile from the context menu.
- Right-click the StartServer node again and chose Execute from the context menu.

  This starts the Enhydra server from within the IDE.
- When Enhydra server is finished booting, access the application with a browser at the following URL:

  `http://localhost:8002`

  You should see the Welcome.html page with a redirect and a dynamic time stamp.

Chapter 7. Using DODS

You can start dods in Dods Generate window with one of the following parameters:

- [doml]

  The doml input file describing data object mapping. Required: Yes
- [outdir]

  Target for generated classes, expressed as a directory path. Required: Yes
- [task]

  Name of Ant task from generate.xml. Required: No.

  This parameter can have one of the following values:

  - without parameters - to create all sql files and java classes and to compile them.
  - dods:sql - to create only sql files.
  - dods:javaNoCompile - to create only java files without compiling them.
  - dods:noCompile - to create SQL files and java files without compiling them.
  - dods:sqlsplit - to create only sql files and separate them in different files using SQLSplitter.
  - dods:noCompileSplit - to create SQL files and separate sql commands using SQLSplitter and java files without compiling them.
- [extensions]

  Template extensions for generating java code. Required: No.

  This parameter can have one of the following values:

  - mdb - generate java code with multi database support.
  - wdwf - generate java code with WebDocWF support.
  - mdbwdwf - generate java code with multi database and WebDocWF support.

Sql and java files will be generated into current project source directory.