

UltraLightClient Installation Guide

UltraLightClient '08 Update 4

canoo

Canoo Engineering AG
Kirschgartenstrasse 5
CH-4051 Basel
Switzerland
Tel: +41 61 228 9444
Fax: +41 61 228 9449
ulc-info@canoo.com
<http://www.canoo.com/ulc/>

Copyright 2000-2009 Canoo Engineering AG
All Rights Reserved.

DISCLAIMER OF WARRANTIES

This document is provided “as is”. This document could include technical inaccuracies or typographical errors. This document is provided for informational purposes only, and the information herein is subject to change without notice. Please report any errors herein to Canoo Engineering AG. Canoo Engineering AG does not provide any warranties covering and specifically disclaim any liability in connection with this document.

TRADEMARKS

Java and all Java-based trademarks are registered trademarks of Sun Microsystems, Inc.

IBM and WebSphere are trademarks or registered trademarks of International Business Machines Corporation.

All other trademarks referenced herein are trademarks or registered trademarks of their respective holders.

Contents

1.1	Overview	1
1.2	Modules and Parts.....	1
1.2.1	Modules	1
1.2.2	Parts.....	2
1.3	Release Structure.....	3
1.3.1	Base Module	3
1.3.2	EJB Module.....	5
1.3.3	Servlet Module	6
1.3.4	Applet Module	6
1.3.5	JNLP Module.....	6
1.3.6	Standalone Module	6
2.1	Getting an Evaluation or Developer Key for ULC	7
2.2	Downloading and Installing the Release.....	7
2.3	Running the Samples Applications	8
2.4	Setting up your Java IDE for Development with ULC	9
2.4.1	System Requirements	9
2.4.2	Basic Setup	9
2.4.3	Importing the Samples and Checking the Installation	12
2.5	License Key Handling.....	16
2.5.1	Deploying ULC Applications.....	16
2.5.2	Upgrading to a New Version	16
2.6	About Milestone Releases	17

Preface

UltraLightClient (ULC) is a library to build Rich Internet Applications (RIA) in Java. Use this standard Java library to develop rich, responsive graphical user interfaces (GUIs) for enterprise web applications within Java EE and Java SE infrastructures.

This version of the *UltraLightClient Installation Guide* accompanies UltraLightClient '08. It gives a general overview of the procedures required to install ULC. It is assumed that the reader is familiar with Java programming and has some familiarity with the Java Swing widget set. Since ULC applications are pure Java applications, they can be built using any suitable Java 2-compatible development environment (JRE 1.4 or higher).

Organization

The Installation Guide is organized into two chapters:

Chapter 1 lists the modules, parts, release structure, and packages of UltraLightClient.

Chapter 2 describes the steps required to install ULC and setup your Java IDE for ULC development.

1 UltraLightClient Release Package

1.1 Overview

The ULC release includes all components required to successfully develop and deploy ULC applications. The following chapter describes the ULC modules and parts. Chapter 1.3 provides an outline of the release structure.

1.2 Modules and Parts

The following section gives a short description of the ULC modules and parts.

1.2.1 Modules

The ULC release is split into modules. Each module belongs to one of five categories:

- **Addon**
Optional addons.
- **Base**
The ULC core.
Always needed, for development and deployment.
- **Container**
Server-side integrations into containers, e.g. Servlet, EJB.
Only needed for server deployment.
- **Environment**
Client-side integrations into environments, e.g. applet, JNLP.
Only needed for client deployment.
- **Sample**
Samples demonstrating the usage of ULC, e.g. *Hello*, *ULCSet*, *OnlineShop*.
Not needed during development or deployment.

The category and module names are reflected in the directory structure of the release, e.g. environment/applet refers to the applet module that belongs to the environment category.

A module directory contains the following subdirectories:

- **Lib**
Contains jar files with module classes.
Sample modules contain the required libraries.
- **Resource**
Only for sample modules: required resources, e.g. images, property files.
- **Src**
Contains either complete source code or source stubs.
Install the source stubs in your IDE to enable code completion.
- **Webapp**
Only for sample modules: ready-to-deploy web application, i.e. war files.

1.2.2 Parts

Each of these modules is composed of up to four parts:

- **Client**
Contains classes to be deployed on the client.
Runs inside the sandbox.
- **Trusted**
Contains classes to be deployed on the client.
Does not run inside the sandbox. Required additional permissions depend on the module.
- **Server**
Contains classes to be deployed on the server side.
- **Development**
Contains all classes needed during development, i.e. the client, trusted, and server parts plus some additional development classes.

The part names are reflected in the filename of the jar files in the `lib` and `src` directory of the corresponding module, e.g. `ulc-applet-client.jar` and `ulc-applet-client-src.jar` for the client side classes of the applet module.

1.3 Release Structure

The current release of ULC has the directory and file structure outlined below.

ulc-2008	UltraLightClient '08 home directory
addon	Addons
qtpintegration	
testframework	
base	ULC base framework
container	J2EE server integration including EJB container integration and Servlet container integration.
ejb	
servlet	
doc	ULC documentation
apidoc	
environment	Client environment integration for applet deployment, JNLP deployment, and standalone deployment.
applet	
jnlp	
standalone	
license	The jar file containing the deployment license key.
previous_releasenotes	Previous release notes for reference purposes.
sample	Sample applications with ready to run Jetty Servlet container.
hello	
onlineshop	
pie	
teammembers	
jetty	
trusted	
ulcdndset	
ulcset	

The modules contain several packages. A description of the packages is offered in the next subsections.

1.3.1 Base Module

To develop a ULC application, you require the application, the development, and the shared packages of the base module. All other packages are intended for ULC extension development only.

Application Packages

The following application packages provide the application-specific components of the ULC framework:

Package <code>com.ulcjava.base.</code>	Contents
<i>application</i>	Server-side ULC widgets as well as application base components
<i>application.border</i>	ULC borders
<i>application.datatype</i>	ULC validators and formatters
<i>application.dnd</i>	Drag-and-drop classes
<i>application.enabler</i>	ULC enablers
<i>application.event</i>	Event classes and listeners to be used with ULC widgets
<i>application.event.serializable</i>	Serializable versions of the listener interfaces in the package above
<i>application.table</i>	Interfaces, models and additional classes used by <i>ULCTable</i>
<i>application.tabletree</i>	Interfaces, models and additional classes used by <i>ULCTableTree</i>
<i>application.tree</i>	Interfaces, models and additional classes used by <i>ULCTree</i>
<i>application.util</i>	Utility classes

Client Packages

The client packages provide the client-specific components of the ULC framework:

Package <code>com.ulcjava.base.</code>	Contents
<i>client</i>	Client-side ULC widget proxies
<i>client.border</i>	ULC borders
<i>client.datatype</i>	ULC validators and formatters
<i>client.dnd</i>	Drag-and-drop classes
<i>client.launcher</i>	Launcher related classes
<i>client.tabletree</i>	All table-, tabletree, and tree-related classes which are not proxies
<i>client.util</i>	Utility classes

Development Package

The development package provides components specific to the ULC development environment.

Package com.ulcjava.base.	Contents
<i>development</i>	Classes making up the development environment

Server Packages

The server package provides the server-specific components of the ULC framework:

Package com.ulcjava.base.	Contents
<i>server</i>	Server base components
<i>server.dnd</i>	Drag-and-drop classes

Shared Packages

These packages provide components of the ULC framework that are common to the application, client, and server packages.

Package com.ulcjava.base.	Contents
<i>shared</i>	Application base components available on the client and on the server
<i>shared.internal</i>	Internally used low-level classes
<i>shared.logging</i>	Classes for the logging infrastructure

1.3.2 EJB Module

This module provides the infrastructure for deploying and accessing ULC applications in an EJB container.

Package com.ulcjava.container.	Contents
<i>ejb.client</i>	Connector to access EJB containers
<i>ejb.server</i>	EJB container adapter bean
<i>ejb.shared</i>	Home and remote interfaces for the bean class

1.3.3 Servlet Module

This module provides the infrastructure for deploying and accessing ULC applications in a servlet container.

Package com.ulcjava.container.	Contents
<i>servlet.client</i>	Connector to access servlet containers
<i>servlet.server</i>	Servlet container adapter class

1.3.4 Applet Module

This module provides the infrastructure for deploying the UI Engine as an applet.

Package com.ulcjava.environment.	Contents
<i>applet.application</i>	Applet content pane for applets embedded in a Web page
<i>applet.client</i>	Applet launcher and services
<i>applet.development</i>	Convenience applet runner for development

1.3.5 JNLP Module

This module provides the infrastructure for deploying the UI Engine using a JNLP client, e.g., Java Web Start.

Package com.ulcjava.environment.	Contents
<i>jnlp.client</i>	JNLP launcher and services

1.3.6 Standalone Module

This module provides the infrastructure for deploying a standalone UI Engine.

Package com.ulcjava.environment.	Contents
<i>standalone.client</i>	Standalone launcher and services

2 Installing ULC

Chapter 2 explains how ULC is installed and describes how to run and install the sample applications. In addition, it includes a short description of license keys and milestone releases.

The tasks involved are:

- Getting an evaluation or developer key for ULC
- Downloading and installing the release
- Running the sample applications provided with the release
- Setting up your Java IDE for development with ULC.
- Importing the samples and checking the installation.

2.1 Getting an Evaluation or Developer Key for ULC

To install ULC a license key is required. Before starting the installation, please order a valid license key from the [product website](#):

- Get an [evaluation key](#) (valid for 30 days), or
- Buy a [developer key](#).

2.2 Downloading and Installing the Release

To install the release:

- Point your browser to: <http://www.canoo.com/ulc/downloads/>
- Download and run the installer corresponding to your platform. The **Introduction** dialog box displays. *Notes for Mac users:*
 1. When using **unzip** to decompress the downloaded file, you might not be able to run the installer. Perform a **chmod -R 777** on the installer application directory and the problem will be solved.
 2. Do not use **open installer.app** as it throws an error. Use double-click.
- Click **Next**.
- Select **I accept the terms of the License Agreement** and click **Next**.
- The **License Information** dialog box displays. Copy the license key you received per email from ---START LICENSE--- to ---END LICENSE--- and paste this key into the **License Information** text box.

Please note: in some cases your email client may introduce line breaks changing the format of the license key email. Please make sure that the each line of the license key text follows the format *name=value*, like in a Java Properties file.
- Click **Next**.
- Select **Typical** to install all ULC libraries and documentaion, or select **Custom** to choose the install set.
- Click **Next**.
- Select the folder in which you would like to install ULC and click **Next** to continue.

-
- Select the shortcuts you would like to create and click **Next**.
 - Review the **Pre-Installation Summary**. Click **Next** to start the installation process.
 - Click **Done** to quit the installer.

Please note: The evaluation key is valid for 30 days. EJB container integration is not activated for the evaluation key. In addition the number of concurrent connections to an application deployed in a servlet container is limited to five.

2.3 Running the Samples Applications

We encourage you to have a close look at the samples. Try modifying, adding new features and experimenting with the samples to learn more about ULC. To do this you will have to set up your Java IDE for ULC development and import the samples into your IDE. The following sample applications with complete source code are part of the release:

- *Online Shop*: multi-step user interaction superior to HTML
- *Team Members*: simple example of master-detail view
- *ULC Set*: ULC standard widget set (equivalent to SwingSet)
- *Pie*: example showing how to extend the ULC standard widget set
- *ULC Drag-and-Drop Set*: Drag-and-drop with ULC
- *Hello*: Hello World

You do not have to set up your Java integrated development environment (IDE) in order to have a first look at these samples.

To run the samples, use the preconfigured Jetty Servlet engine provided with the release:

- Open the **Start** or **Program** menu and select **UltraLightClient '08 > Sample Applications > Start Sample Application Server** to start the server.
- Open the **Start** or **Program** menu and select **UltraLightClient '08 > Sample Applications > Sample Applications Index** to open the sample index page.
- Select a sample application in the **Sample Applications Index** page and click **Start as Applet** to view the applet version or **Start as JNLP** to view the Java Web Start version.

As an alternative, you may also deploy the sample as a Servlet in a Servlet engine of your choice (see [1] for the freely available Jakarta Tomcat servlet engine). The client will be run as an applet or using Java Web Start (see [2]) and connects to the application using the HTTP protocol.

To deploy the sample as a Servlet:

- Install the application as a Servlet. Each sample is packaged as a web application archive (*.war*) file (see [3]) containing the application classes and all necessary libraries. The archive contains HTML files to start the client and the classes for the UI Engine (to be downloaded as an applet or using Java Web Start).
- Deploy the web application archive file *<sample name>.war* in the *sample/<sample name>/webapp* directory into your Servlet container. See your Servlet container's instructions on how to deploy web applications (for the Jakarta Tomcat Servlet engine simply copy the file into the *webapps/* directory).

-
- Start your Servlet engine, browse to `http://localhost:<port>/<sample name>/index.html` and click on the applet hyperlinks or on the JNLP hyperlink. For the latter to work, you must have Java Web Start or any other JNLP client installed.

2.4 Setting up your Java IDE for Development with ULC

This section describes how to set up your Java IDE for development.

2.4.1 System Requirements

You may use any Java IDE that provides support for the Java 2 platform (JDK 1.3 or later).

2.4.2 Basic Setup

Please find listed below a step-by-step description how to set up Eclipse for development with ULC. This section lists the jar files required for the basic setup.

To set up your Java IDE for development with ULC, start the IDE and add the ULC libraries to the classpath of your IDE:

```
base/lib/ulc-base-development.jar
container/servlet/lib/ulc-servlet-client.jar
container/servlet/lib/ulc-servlet-server.jar
container/ejb/lib/ulc-ejb-client.jar
container/ejb/lib/ulc-ejb-server.jar
environment/applet/lib/ulc-applet-development.jar
environment/jnlp/lib/ulc-jnlp-client.jar
environment/standalone/lib/ulc-standalone-client.jar
```

This is the basic setup. See the next section for a detailed description. For further details see *Development Environment* in the [ULC Reference Guide](#) and the sections on *Client Deployment* and *Server Deployment* in the [ULC Deployment Guide](#) for further details.

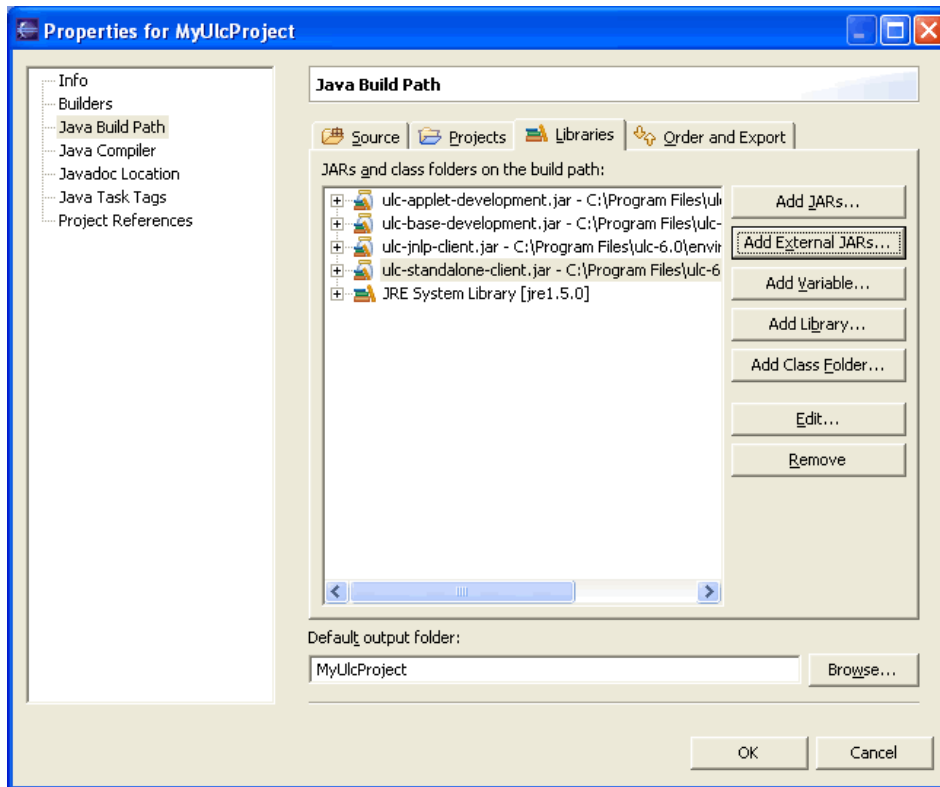
Setting up Eclipse for Development with ULC

As mentioned in Chapter 2.4.1, you may use ULC with any Java IDE. This section describes how to set up ULC for development within the Eclipse IDE.

To set up Eclipse:

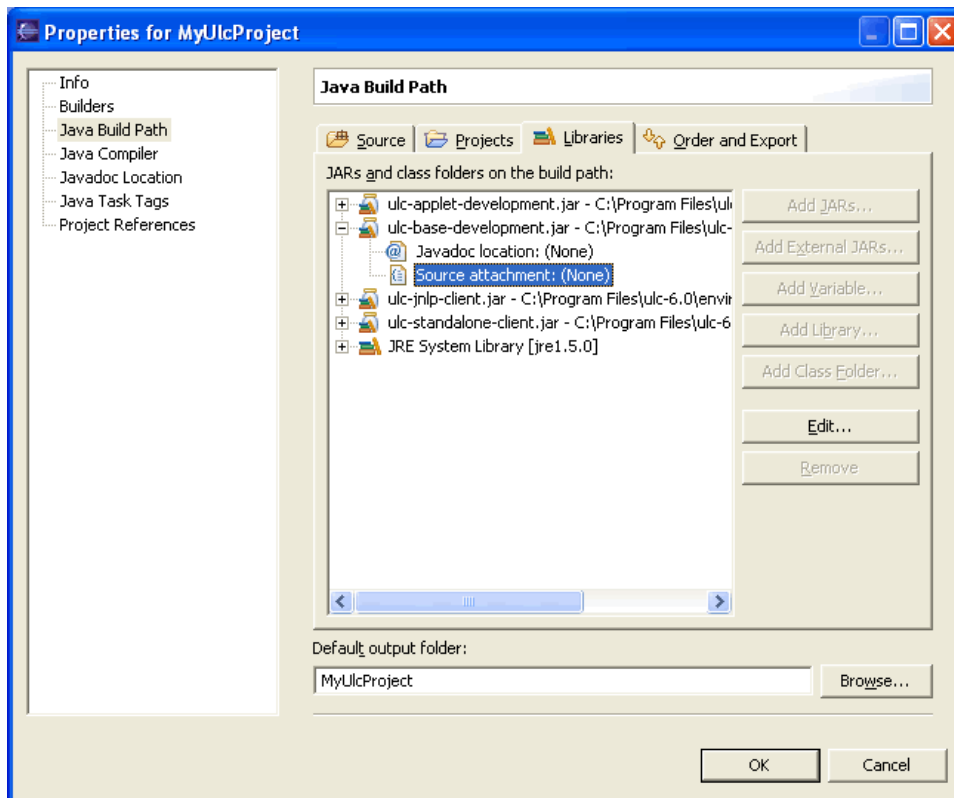
- Start Eclipse
- Select **File > New > Project > Java Project**.
- Type the project name, e.g. *MyUlcProject*, and click **Finish**.
- Right-click the project name in the **Package Explorer** and select **Properties** in the menu.

- Select Java Build Path > Libraries.



- Click **Add External JARS**.
- Browse to the corresponding lib folder to select a lib jar. Repeat this for all ULC libraries. The corresponding libraries are located in:
`base/lib/ulc-base-development.jar`
`container/servlet/lib/ulc-servlet-client.jar`
`container/servlet/lib/ulc-servlet-server.jar`
`container/ejb/lib/ulc-ejb-client.jar`
`container/ejb/lib/ulc-ejb-server.jar`
`environment/applet/lib/ulc-applet-development.jar`
`environment/jnlp/lib/ulc-jnlp-client.jar`
`environment/standalone/lib/ulc-standalone-client.jar`

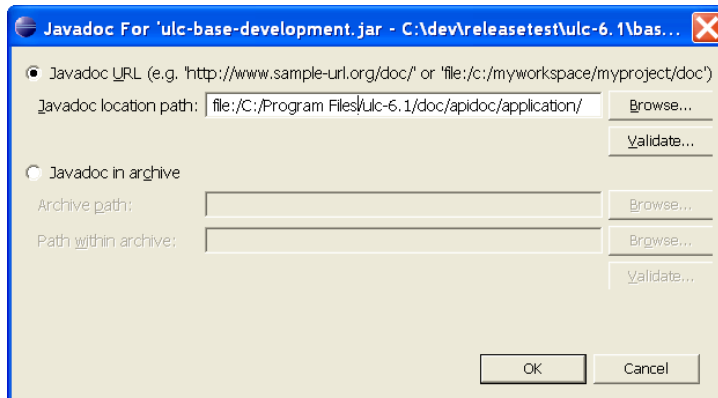
- Next, add the source stubs for code completion to the **Java Build Path**. Expand the node of a ULC library in the **Libraries** tab.



- Select **Source attachment: (none)** and click **Edit**.
- Click **External File** and browse to the corresponding `src` folder to select the `src` jar. Repeat this for all ULC libraries. The corresponding sources are located in:
 - `base/src/ulc-base-development-scr.jar`
 - `container/servlet/src/ulc-servlet-client-scr.jar`
 - `container/servlet/src/ulc-servlet-server-scr.jar`
 - `container/ejb/src/ulc-ejb-client-scr.jar`
 - `container/ejb/src/ulc-ejb-server-scr.jar`
 - `environment/applet/src/ulc-applet-development-scr.jar`
 - `environment/jnlp/src/ulc-jnlp-client-scr.jar`
 - `environment/standalone/lib/ulc-standalone-client.jar`
- To add Javadoc for the corresponding libraries, expand the node of a ULC library in the **Libraries** tab.
- Select **Javadoc location: (none)** and click **Edit**.

- Javadoc is located in:
`doc/apidoc/application`

Click **Browse** and select the **apidoc** folder.



- Click **OK**.
- Repeat this for all ULC libraries.
- Click **OK** to close **Properties for MyULCProject**.

Your Eclipse environment is now ready for development with ULC.

Please note: the [ULC Code Community](#) offers a useful utility plug-in for Eclipse. See [Eclipse IDE Integration](#).

For details on how to install ULC Visual Editor for Eclipse, please see the [Developer Zone](#) or consult the online help included in the release.

Finally, add the source code for the samples included in the release. The next section describes how to import the samples.

2.4.3 Importing the Samples and Checking the Installation

In order to run the samples within your IDE, follow the setup procedure as described in Section 2.4.2 and then import the the samples into your IDE.

To import a sample:

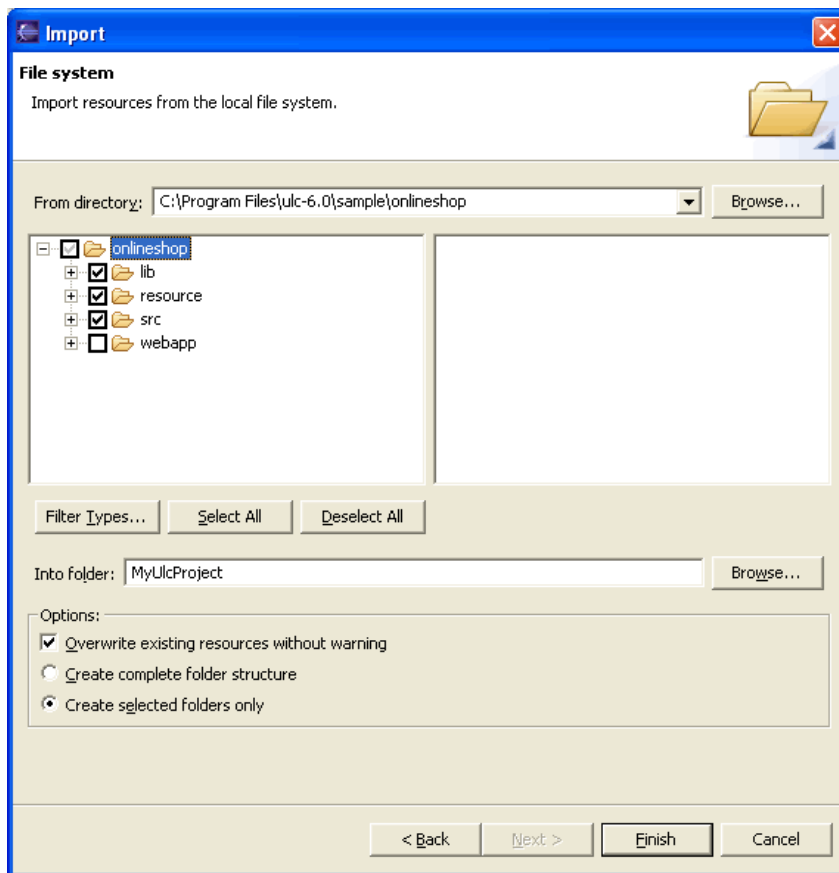
- Import the *lib*, *resource*, and *src* directories.
- Add the files in the *src* folder as sources.
- Add the *resource* folder to the class path.
- Add the jar files in the *lib* folder to class path.
- Compile the samples and then run the samples by executing the respective main classes, e.g., `com.ulcjava.sample.hello.Hello`.

If the samples compile and run without errors, your setup is correct and you can start developing your first ULC application. The next section provides a detailed description how to import the *OnlineShop* sample into Eclipse.

Importing the OnlineShop Sample into Eclipse

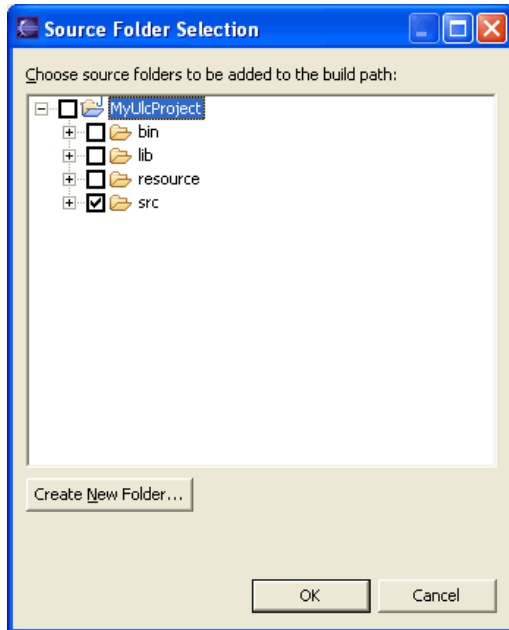
The following description shows how to import the *OnlineShop* sample application into Eclipse:

- Select your ULC project, in our example: *MyULCProject*.
- Right-click the project name in the **Package Explorer** and select **Import** in the menu.
- Select **File System**.
- Browse to the location of the *OnlineShop* sample and select the **onlineshop** folder.
- Click **OK**.

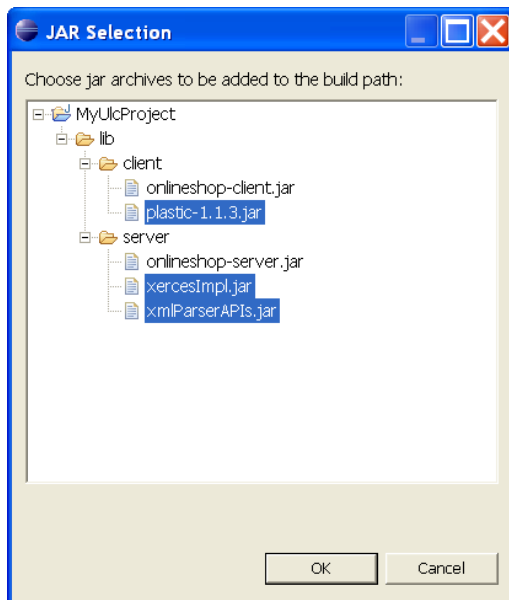


- Expand the node and select **lib**, **resource**, and **src**.
- Click **Finish**.
- Right-click the project name in the **Package Explorer** and select **Properties** in the menu.

- Select **Java Build Path > Source > Add Folder...**

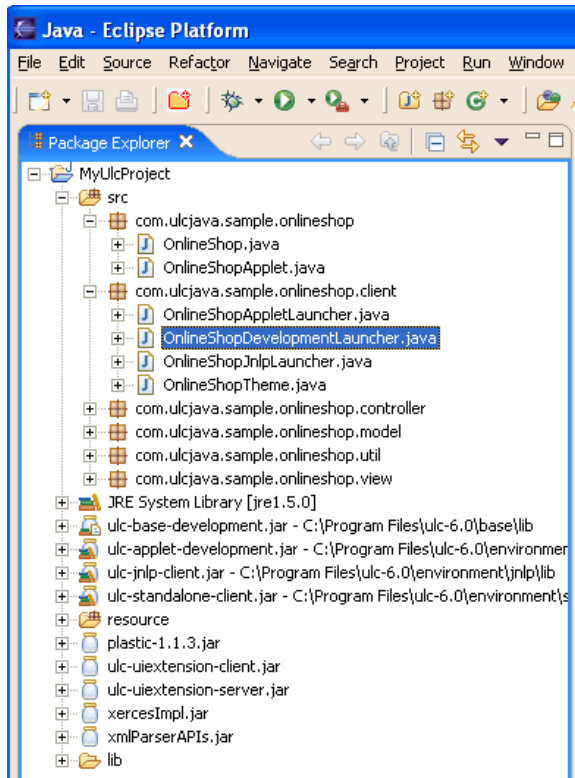


- Select **src** and click **OK**.
- Click **Yes**, if the dialog box **Source Folder Added** displays.
- To add the resources, select **Libraries > Add Class Folder**.
- Select **resource** and click **OK**.
- To import the *lib* directory, select **Libraries > Add JARS...**



- Expand all nodes in all folders.
- Select the following files:
plastic-1.1.3.jar
xercesImpl.jar
xmlParserAPIs.jar
- Click **OK**.

- Click **OK** to close **Properties for MyULCProject**.
- The **Setting Build Path** dialog displays. Click **Yes**.



- Select **OnlineShopDevelopmentLauncher** in the Package Explorer.
- Right-click and select **Run > Java Application** to run the OnlineShop sample within Eclipse.

2.5 License Key Handling

To develop with ULC, a developer key or evaluation key is required. To buy a developer key, please see <http://www.canoo.com/ulc/BuyNow> or contact [Canoo Engineering AG](#) directly. To receive an evaluation key, please register at <http://www.canoo.com/ulc/Registration>. The evaluation key is valid for 30 days.

To install the evaluation or developer key, copy and paste the license key into the **License Information** text box displayed during installation. See the detailed description in Chapter 2.2.

The following table provides an overview of the license types available for ULC:

Type of Key	Description
Evaluation Key	Restricted license, for evaluation purposes only, valid for 30 days. Limited to 5 concurrent connections per web container. Not included: EJB container adapter. No sublicensing or commercial deployment allowed. To run an application, include the jar file /license/ulc-deployment-key.jar in the deployment.
Developer Key	Required to develop applications with ULC. To run an application, include the jar file /license/ulc-deployment-key.jar in the deployment.
Upgrade	If you purchased a developer key within 90 days before the new release was announced, or if you have a valid Maintenance license associated with your developer key, you are entitled to a free upgrade. Customers without Maintenance, please see the License and Pricing Model or contact Canoo for upgrade pricing.

2.5.1 Deploying ULC Applications

To deploy an ULC application, the .jar file `/license/ulc-deployment-key.jar` must be included in the application deployment.

For further details on deploying ULC applications, please refer to the [ULC Deployment Guide](#).

2.5.2 Upgrading to a New Version

If you purchased a ULC developer key within 90 days before a new release of ULC is made available, or if you have a valid ULC Premium Support license associated with your developer key, you are entitled to a free upgrade key.

To upgrade, type your current developer key into the field provided at <http://www.canoo.com/ulc/upgrade>. If you are entitled to a free upgrade key, a new developer key will be sent to your email address.

2.6 About Milestone Releases

Canoo offers early access to new features and fixed problem reports. These releases are referred to as milestone releases. Milestone releases are not official releases and are provided for evaluation and testing purposes only. The milestone release notes provide an overview of fixed problem reports and added feature requests.

References

-
- [1] Jakarta Tomcat servlet container
<http://jakarta.apache.org/tomcat/>
 - [2] Java Web Start
<http://java.sun.com/products/javawebstart/>
 - [3] J2EE tutorial on web application archives
<http://java.sun.com/j2ee/tutorial/doc/WCC3.html>
 - [4] Eclipse
<http://www.eclipse.org/>
 - [5] Jakarta ORO regular expression package
<http://jakarta.apache.org/oro/>
 - [6] HTML 4.0.1 specification
<http://www.w3.org/TR/html4/>
 - [7] Java Secure Socket Extension (JSSE) 1.0.3
<http://java.sun.com/products/jsse/index-103.html>
 - [8] Java Security Architecture
<http://java.sun.com/j2se/1.3/docs/guide/security/index.html>
<http://java.sun.com/j2se/1.4.1/docs/guide/security/index.html>
 - [9] JNLP forum thread describing the protocol handler workaround
<http://forum.java.sun.com/thread.jsp?forum=38&thread=71335>
 - [10] Java Plug-In tags
<http://java.sun.com/products/plugin/1.3/docs/tags.html>
<http://java.sun.com/products/plugin/versions.html>
 - [11] Using the conventional APPLETTAG with Java Plug-In 1.3.1 and 1.4
http://java.sun.com/products/plugin/1.3.1_01a/faq.html
http://java.sun.com/j2se/1.4/docs/guide/plugin/developer_guide/applet_tag.html
 - [12] "usePolicy" Permission
<http://java.sun.com/products/plugin/1.3/docs/netscape.html#use>
 - [13] LiveConnect
http://developer.netscape.com/docs/technote/javascript/liveconnect/liveconnect_rh.html
 - [14] Calling an applet from Java Script
<http://java.sun.com/products/plugin/1.3/docs/jsobject.html>.
 - [15] InstallAnywhere
<http://www.installanywhere.com/>
 - [16] InstallShield
<http://www.installshield.com/>
 - [17] Initializing an initial context with applet parameters
<http://java.sun.com/j2se/1.3/docs/api/javax/naming/Context.html#APPLET>
 - [18] Canoo Engineering AG contact address
ulc-info@canoo.com
 - [19] Introduction to Servlet technology
<http://java.sun.com/products/servlet/index.html>
 - [20] Servlet specifications
<http://java.sun.com/products/servlet/download.html>

-
- [21] Introduction to EJB technology
<http://java.sun.com/products/ejb/index.html>
 - [22] EJB specifications
<http://java.sun.com/products/ejb/docs.html>
 - [23] J2EE Tutorial
<http://java.sun.com/j2ee/tutorial/index.html>
 - [24] Packaging J2EE applications
<http://java.sun.com/j2ee/tutorial/doc/Overview4.html>