HDCOOKBOOK README

This directory contains folders that are used to produce a set of applications (also called “xlets”) on the disc included with the HD Cookbook, tools of general use for BD-J development, and documentation and other assets. There is a great deal of sharing of source code between the three xlets, and between the xlets and some of the tools that are used to produce the final xlet bundles. Each application picks and chooses the packages that it needs as part of the compilation process.

> **IMPORTANT NOTE FROM JANUARY 2022**: These tools work with JDK 1.8. As of this writing, JDK 1.8 is still supported, and it looks like it will be supported for a long time. Some of the code here depends on “non-public” classes in the sun namespace. These classes are known to exist in the OpenJDK implementations available at this time. This has been tested with the Azul JDK 1.8 implementation on MacOS.
>
> The `hdcookbook_discimage` demo has been tested, and works. Note that the `HDCookbookDiscImage.zip` file produced by the build script must be combined with the non-generated assets for the disc image, found at https://github.com/zathras/java.net/releases/download/1.0.1/hdcookbook_disc_image_without_java.zip.
>
> If memory serves, the last version of the cookbook tools released to java.net was 1.1. FWIW, I’ll call this release from 2022 that works with JDK 1.8 “version 1.2”.

To see how to build the source, skip down to the “build.xml” entry.

The HD Cookbook source was originally at https://hdcookbook.dev.java.net. It has since been moved to https://github.com/zathras/java.net/ under hdcookbook.

The main directories to look in are DiscCreationTools and AuthoringTools. Each has a README that describes the sorts of things you’ll find there. For each distinct tool or library, we try to be good about having at least a README file in the base directory that talks about what it does, and there’s a good amount of documentation in the form of javadoc comments in the various source files. With a bit of digging, we hope you’ll find what you need.

Following is a non-exhaustive list of some of the highlights, concentrating on what is needed to build the “bookmenu” disc image that was the central sample BD-J disc in the HD Cookbook. There are many interesting tools that are *not* listed here, so be sure to poke around the DiscCreationTools and AuthoringTools directories.

> **DiscCreationTools/bdjo**

This directory contains to tool that can read and write BD-J Object, or BDJO files. A BDJO file tells the player where to look for the xlets and other resources that control disc playback. The BDJO tool in this directory will read a XML file containing a representation of the BDJO structure, and emit a binary .bdjo file suitable for inclusion on a disc. The code in this directory is stand-alone, but it does require JDK 1.6 to run.

DiscCreationTools/security
This directory contains a JAR signing tool for BD-J xlets.

Xlets/hdcookbook_discimage/gunbunny
This contains the "Gun Bunny" game xlet. The java code is in this directory and the "util" subdirectory, whereas the images are in the assets subdirectory. Gun Bunny is a stand-alone xlet; it does not depend on other packages in com.hdcookbook. However, Gun Bunny uses the same "Lisa" font that the menu xlet uses. That font is stored with the menu xlet's assets.

Xlets/hdcookbook_discimage/bookmenu
This contains the two xlets that make up the menuing system on the HD cookbook disc image. The src directory has the following components:

> **com/hdcookbook/bookmenu**
> This directory contains one interface definition that is shared between the monitor xlet and the menu xlet.
>
> **com/hdcookbook/bookmenu/monitor**
> This directory contains the code for the monitor xlet. The monitor xlet is a small xlet that manages the other two xlets. In our disc, the monitor xlet launches the game and terminates the menu when the menu xlet requests that it do so. When the game ends, it re-launches the menu xlet. The monitor xlet uses a small utility class from GRIN (com.hdcookbook.grin.util.Debug). In this directory you'll also find the permission request file (PRF) for the monitor xlet. This is used in the codesigning process.
>
> **com/hdcookbook/bookmenu/assets**
> This directory contains the non-code assets used by the menu xlet. The file "menu.txt" contains the GRIN script that defines most of the visual user interface of the menu, and much of its behavior. This file is parsed at xlet startup by the GRIN framework. Also in this directory are the main graphics in the Graphics subfolder, the sound effects in sound.bdmv, and the font used by the menu and by Gun Bunny, in Font. The images in the Graphics directory are compiled by the *mosaic builder* program in GRIN to form two image mosaics that are used in the original disc image. The actual

images themselves are not present in the disc image.

In this directory you'll also find a directory called "FakeNetwork." This is used to simulate loading an updated bio for Gun Bunny on players that don't have a network connection.

com/hdcookbook/bookmenu/menu (and subdirectories except "test_assets")
In this directory, you'll find the Java code for the menu xlet, as well as a permission request file (PRF) used for code signing. This xlet relies heavily on the GRIN framework to present graphical and sound assets, and to handle remote control and mouse input.

com/hdcookbook/bookmenu/menu/test_assets
This contains a PNG image that can be used when testing the menu's GRIN script. This PNG is not copied the disc image. It's used with a GUI emulator tool that's part of the GRIN framework. That tool can produce a snapshot of a simulation of the UI state of the xlet at a given point of time. That snapshot looks better if you feed the tool a PNG image that simulates background video; that's what's in this directory.

Xlets/hdcookbook_discimage/bdjo
This contains the XML definition of the BDJO file on our disc. It is compiled by com.hdcookbook.tools.bdjo into a binary BDJO file by one of the build scripts.

AuthoringTools/grin/library
This directory contains code and documentation for GRIN, a framework for graphical interactivity. GRIN is really the heart of the menu xlet. GRIN is described in the file doc-files/index.html.

AuthorinTools/grin/jdktools/mosaic
This directory contains a program that runs on JDK 1.5 (or higher) that automatically combines images into an image mosaic. It reads one or more GRIN show files and locates all of the images used in that file. It then creates an efficient set of image matrixes along with a companion binary metadata file that can be used to reproduce the contents of the original source images. The GRIN framework reads both the matrix and the metadata file at xlet start up time. The desktop java mosaic builder program uses the GRIN framework itself fairly extensively, so when this tool is compiled it includes all of GRIN.

AuthoringTools/grin/jdktools/test

This directory contains three programs (and assorted other classes) used

to test and exercise the GRIN framework, and to test and debug a GRIN show. They run on desktop java (JDK 1.5 or higher). Desktop Java is affectionately known as “big JDK” in some circles. The first program, GrinTestRyan, is of mostly historical interest. It runs a simulation of “Ryan’s Life” on big DJK. The second program is “GenericMain”. It lets you read and display any show file. It shows the show file in a Frame, but the interface to control the show is a command-line interface, so you have to run this from a terminal window.

The most interesting program in this directory is GuiGenericMain. It’s like GenericMain, but it adds a GUI interface to control a show file. With GuiGenericMain, you can read in any valid GRIN show. You’re presented with a tree widget that shows all of the segments in the show. By double-clicking a segment, you tell GRIN to move the show to the given segment. In this way, you can simulate the behavior of a GRIN show file on desktop Java, and see what it looks like. This program also lets you set the frame rate, step through a GRIN animation frame-by-frame, or even take a snapshot of the UI’s state and output a PNG image.

In the test directory, you’ll find an old GRIN show file and image assets that were used in the first xlet to use GRIN. It was called “Ryan’s Life,” and was part of a demo developed by Sun and Technicolor. Ryan’s Life is not used in any way in the Blu-ray disc image in the HD cookbook, but it’s of some historical interest with GRIN, and it’s useful for testing changes to GRIN. However, the menu xlet in com.hdcookbook.bookmenu.menu is a much better guide to how to create a GRIN show than the original Ryan’s Life demo contained here.

vars.properties
This is a configuration file for the ant build environment. See "build.xml" for details.

build.xml
This is an ant build file. Ant is a powerful java-based build tool, available from ant.apache.org. Run “ant” to build a HD cookbook sample image, or “ant all” to build everything in the repository. Before building, there are some variables that need to be set up for your environment, which can be done by editing vars.properties file in the same directory. One thing you’ll need to come up with is a classes.zip file that contains the signatures of the BD-J platform; we don’t have permission to redistribute that ourselves.

> ***WARNING***: To compile, you’ll need a classes.zip containing the signatures of the BD-J platform. It is not provided in the HD cookbook source repository. To create a copy of this necessary resource, see “How can I get a BD-J Platform Definition” at http://wiki.java.net/bin/view/Mobileandembedded/Blu-RayDisc

Build/shell
This directory contains some convenience shell scripts. The two whose names start with “build” just invoke the ant build system. make_grin_javadocs.sh re-builds the javadoc tree for the GRIN library. run_book_menu runs the GRIN show viewer described under jdktools/test on the assets of the HD cookbook’s main menu. This is a nice way to play with GRIN on desktop java.

By the way, for a Unix-like shell on windows, I’ve had good luck with zsh from http://unxutils.sourceforge.net/. Note that the “sh.exe” that comes with unxutils is actually zsh, so it reads its startup commands from ~/.zshrc (where ~ is c:/Documents and Settings/userid). For anything other than casual use, you’ll probably want to set up a real build environment using your IDE of choice.

www
This contains the website at https://hdcookbook.dev.java.net. That website also contains a SVN (subversion) repository that contains the latest version of the src director. It will either contain the “scripts” directory, or something better, or both :-) Under this directory, you can find the generated javadocs of GRIN, which can be handy.