

Getting Started with the Livescribe™ Platform SDK

Livescribe™ Platform SDK
Version 1.5

Copyright and Trademarks

LIVESCRIBE, ECHO, PULSE, and PAPER REPLAY are trademarks or registered trademarks of Livescribe, Inc. Anoto is a trademark of Anoto Group AB. All other brand and product names are trademarks of their respective owners.

Copyright © 2007-2010 Livescribe Inc. All rights reserved.

GettingStartedPlatformSDK-SDK-1.5.0-REV-E

12/20/2010 12:08 PM

Table of Contents

Copyright and Trademarks	ii
New Features and Improvements	1
About Livescribe Eclipse Features	1
Livescribe Projects in Eclipse	2
Livescribe Perspectives in Eclipse	2
Penlet Perspective	3
Penlet Feature.....	4
Paper Design Perspective.....	5
Penlet JARs and Bundles	6
A Basic Tap-and-Play Application.....	7
Programming a Hello World Penlet.....	12
Using the Livescribe Penlet Project Creation Wizard	12
Programming a Simple Hello Open Paper Penlet	19
The strokeCreated Method	24
The canProcessOpenPaperEvents Method	26
Designing a Simple Paper Project	26
The Livescribe Paper Project Creation Wizard	26
Programming a Fixed Print Application	28

Design and Create Page Images.....	29
Create a Penlet Project	29
Create a Paper Product Project	29
Import the Livescribe Flip Notepad Insert Card AFD File	30
Link the Paper Product to the Penlet.....	31
Create Active Regions (Shaping).....	32
Examine the Livescribe Standard Controls.....	33
Auto-generate Event Code for Your Active Region	35
Complete the Penlet Programming	36
Import the Hello World Audio File	36
Set Up the Media Player	37
Edit the PenDown Method	38
Print and Test the Paper Product and Penlet	38
Penlet Configuration Editor	40
Properties Tab	41
Image Resources Tab	41
Audio Resources Tab	42
Text Resources Tab.....	43
Advanced Tab	44
Importing a Penlet Project.....	45

Running and Debugging with Livescribe Smartpen Emulator (Windows Users)	46
Setting Up the Smartpen Emulator	46
Running a Penlet using the Smartpen Emulator	47
Debugging a Penlet using the Smartpen Emulator	48
Recommended Settings for Emulator Performance	50
Using the Smartpen Debug View	50
Viewing the Debug List	52
Smartpen Info View	53
Setting Up Library Projects in Eclipse	54
Using the Livescribe Context Menu	54
Add/Remove Event Handlers	56
Add Paper Area Handler	57
Remove Paper Area Handler	58
Rename Paper Area Handler	58
Add Images or Sounds	58
Add System Sounds	59
Using the Livescribe Penlet Project Creation Wizard	60
Penlet Project Creation Page	61
Event Configuration Page	63
Initialize Text in Configuration File	65

Intelligent Character Recognition (ICR) Configuration Page	66
Application Configuration Page.....	68
Preparing Smartpen Resources	70
Adding Smartpen Display Images.....	70
Editing Smartpen Display Images using the ARW Editor	71
Converting Audio	71
Building and Deploying Applications.....	72

New Features and Improvements

Livescribe made feature additions and other improvements to the Platform SDK, including:

- A new Tap-and-Play sample application that uses name matching between the penlet resources and paper product area names. No java programming is required to use this sample code.
- Support for Paper Tablet functionality in custom paper products. You can create areas on your paper products that can be used as tablet regions for the Paper Tablet application.
- Install and launch your penlet automatically with the Smartpen Emulator using the new Run as > Penlet feature and the Debug as > Penlet feature added in version 1.4. (Windows only)
- Ability to view a smartpen's full debug list (a list of the most recent debug RAMs).
- Ability to create installation bundles (.bnd files) with group names and versions. In the future, this feature will help streamline submitting applications to Livescribe.
- Ability to set up library projects to organize your code using Eclipse projects.
- Improvements to the ARW image editor including support for full-size images (256x256) and provides drawing tools for ellipses, rectangles, and lines.
- Ability to disable tooltips for the Paper Project editor.
- Ability to add smartpen display images and audio files directly from Penlet Configuration Editor. The editor launches the audio converter and image converter automatically.

About Livescribe Eclipse Features

Livescribe plugins for Eclipse are grouped as two Eclipse features:

- Livescribe Penlet SDK feature
- Livescribe Paper Designer feature

Getting Started with the Livescribe Platform SDK

Each feature is designed for a particular kind of project. The Livescribe Penlet SDK feature allows you to create a penlet project. The Livescribe Paper Designer feature allows you to create a paper project.

The **File > New** menu contains two items that pertain to development on the Livescribe Platform.

Menu Item	Action
Livescribe Penlet Project	Launches the Livescribe Penlet Project Creation Wizard.
Livescribe Paper Project	Launches the Livescribe Paper Product Wizard. See The Livescribe Paper Project Creation Wizard.

Livescribe Projects in Eclipse

In Eclipse, a *workspace* manages one or more *projects*. A project is typically a folder structure containing many files that make up an application.

A Livescribe *paper project* is used to store files associated with the design and creation of Livescribe dot paper. A Livescribe *penlet project* is used to store code and resources that are compiled into a penlet.

The paper you create typically has a dedicated Java penlet associated with it that interacts with pre-defined active regions on the paper. Each paper project contains one principle file called an AFD, as well as some auxiliary files. The AFD is a proprietary document format, similar to a zip archive, which contains files and information about a given paper product.

Livescribe Perspectives in Eclipse

Eclipse uses perspectives to help organize projects. A *perspective* is a collection of *views* (windows) and associated *actions* (toolbar buttons and menu options) that are required for a particular kind of project.

The Livescribe Eclipse features provide two perspectives: a Penlet and a Paper Design perspective. Each one contains all views and actions you need to use the

feature. You display a perspective by selecting **Window > Open Perspective**, and then selecting **Paper Design** or **Penlet** from the popup menu.

Note: The text editor, which contains the source code of a penlet, is the same in either perspective. When you change perspectives, the text editor will continue to display the same source files.

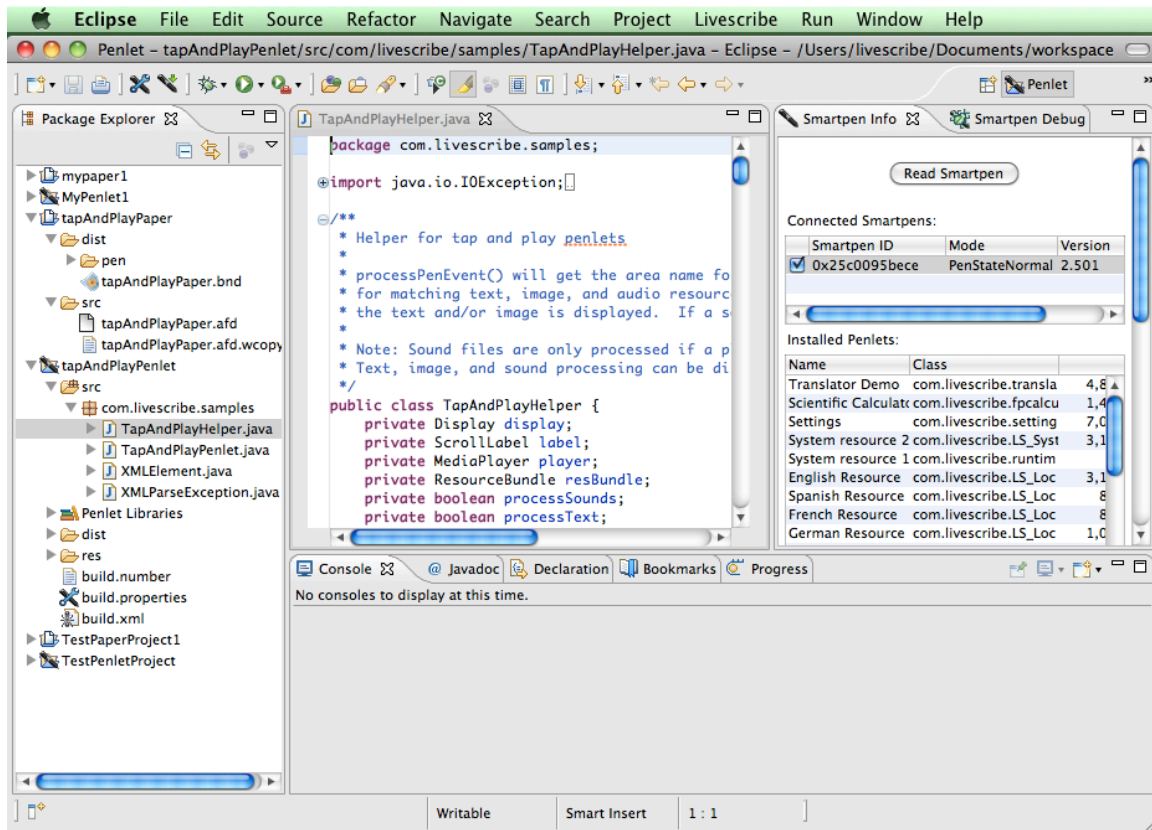
Penlet Perspective

The Penlet Perspective contains these capabilities:

- **Actions** - Toolbar buttons and menu options allow you to add events and handlers, deploy penlets to a smartpen, convert and edit resources, and more. To configure penlet resources, use the Penlet Configuration Editor.
- **Package Explorer:** a tree control that displays the projects in the workspace. The name of each project appears on the top-most level. Click the plus next to a project name and drill down to see the various files that make up the project. Although penlet and paper projects appear in this view, Package Explorer is designed to drill down into Java projects such as penlets. To drill down into paper projects, switch to the Paper Design perspective.
- **Views** - two views, Smartpen Info and Smartpen Debug, which provide information about the state of the connected smartpen.

Note: The views window may be so narrow that the full title is not visible (such as **Smar** instead of **Smartpen Debug**). Expand a view by clicking and dragging the left border.

Getting Started with the Livescribe Platform SDK



Penlet Feature

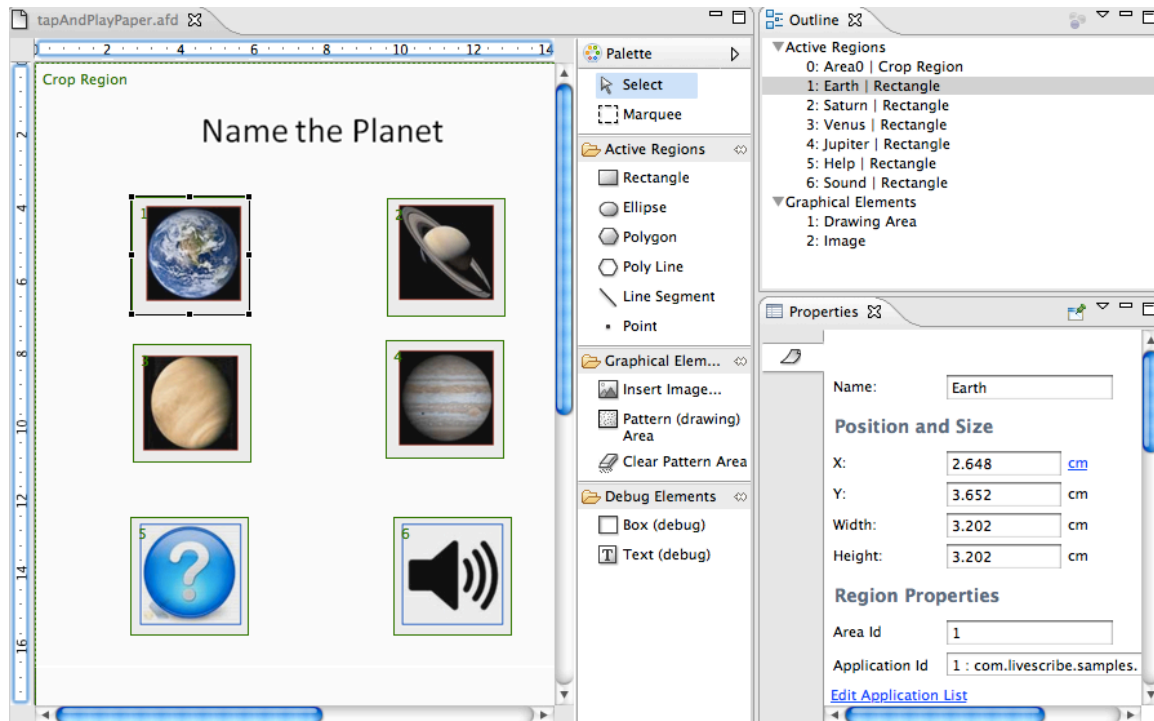
The Penlet feature contains these capabilities:

- **Livescribe Penlet Project Creation Wizard** - The project wizard simplifies the process of creating penlet projects.
- **Image Editor** - a simple editor for setting pixel values in a smartpen-compatible ARW file.
- **Import Project Wizard** - The SDK makes it possible to import penlet projects created outside of the SDK.

For detailed information on using the Penlet feature, please see [Programming a Hello World Penlet](#).

Paper Design Perspective

The Livescribe Paper Design Perspective is optimized for working with paper projects. You will be asked to switch to this perspective when creating a new paper project.



The views used by the Paper Design perspective are:

- **Thumbnails:** a list of page images, providing a quick way to navigate the pages of the document.
- **Project Explorer:** a tree control that displays the projects in the workspace. The name of each project appears on the top-most level. Click the plus next to a project name and drill down to see the various files that make up the project. Both penlet and paper projects will appear in this view.
- **Outline:** a list of all the graphical elements and active regions defined on the page
- **Properties:** properties of the currently selected item.

The Paper Design feature is a very powerful tool. For detailed information on using it to create paper products, refer to the *Developing Paper Products* guide in the Livescribe Platform SDK installation.

Penlet JARs and Bundles

As a developer, you deploy Penlets to smartpens as Java archive (JAR) files. Similarly, you deploy Paper Products to smartpens as AFD files.

Note: Livescribe deploys applications to end users as digitally-signed bundles (.bnd files). Although you can build bundles using the SDK, you cannot deploy them without prior approval from Livescribe.

Each JAR contains a Java class that extends the `Penlet` class, and accompanying classes and resources. The contents of a penlet JAR is similar to regular Java archives, but there are a few significant differences:

- Penlet classes are based on the Java Platform, Micro Edition (J2ME), specifically the Connection-Limited Device Configuration (CLDC version 1.1). This means penlet classes cannot access all of the libraries supplied with standard Java.
- Like all J2ME CLDC classes, classes in a penlet JAR must be pre-verified. The off-pen pre-verification step checks the classes for safety and security, and thereby reduces the processing performed by the smartpen's JVM (Java Virtual Machine).
- Penlet JARs contain resources such as audio or image files that can be played or displayed on the smartpen. These resources will be discussed in a later section.
- The JAR that Eclipse builds for a project can be found in the project's `dist/lib` folder in the Project Explorer.

A Basic Tap-and-Play Application

A “Tap and Play” application consists of a penlet and a paper product with button-like active regions. A smartpen responds to these regions by playing a sound and/or displaying text or an image on its screen.

To make creating these applications as simple as possible, the SDK provides sample code that you can use supplying resource files whose names match the corresponding regions.

Note: *There is no penlet coding required if you use the penlet provided with this sample. However, you can subclass or modify this sample to implement additional functionality.*

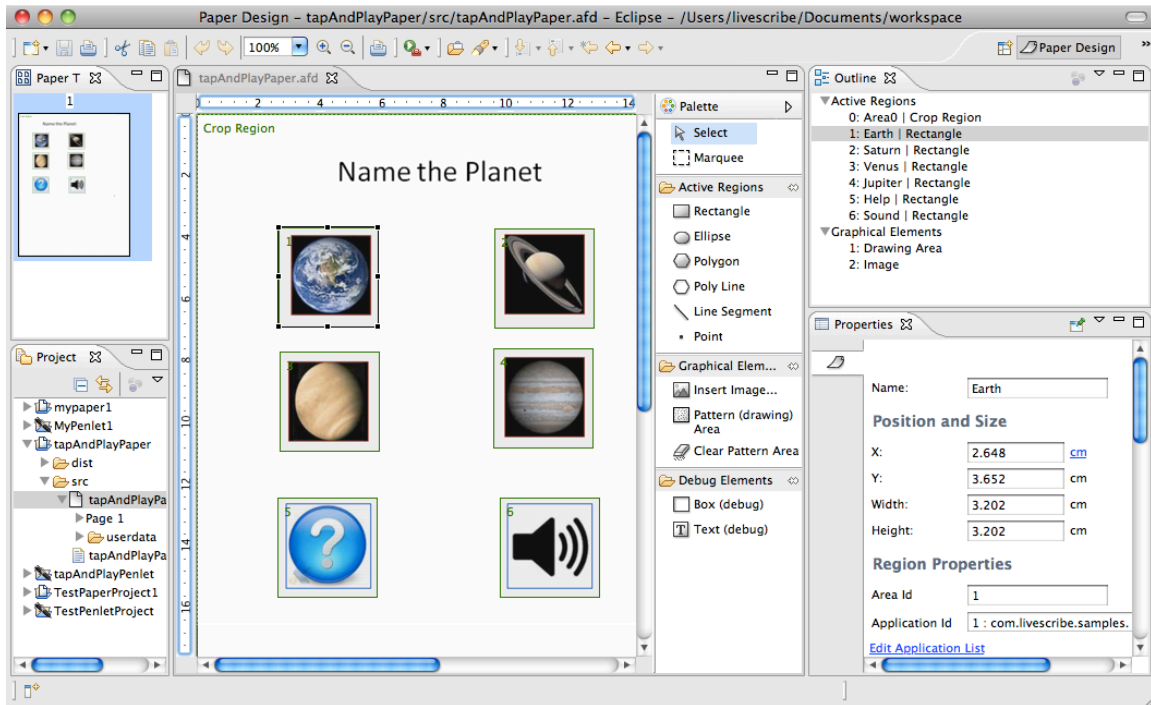
The tap-and-play sample code maps the active area names to corresponding audio files and text strings. The key is to have the area names match the resource file that you want to play or display.

For example, if you add a new rectangle to the paper, your area might default to the name Area7. If you add an audio file named Area7.wav then that file will automatically play when the rectangle is tapped.

The sample provided is a simple Name the Planets application. For example, tapping on the area named “Earth” plays the audio file named Earth.wav and displays a text string about Earth.

Using this sample application, you can build your own Tap-and-Play paper product and working penlet. As an example, the following steps describe how to change the Name the Planets application to a Name the Animals application.

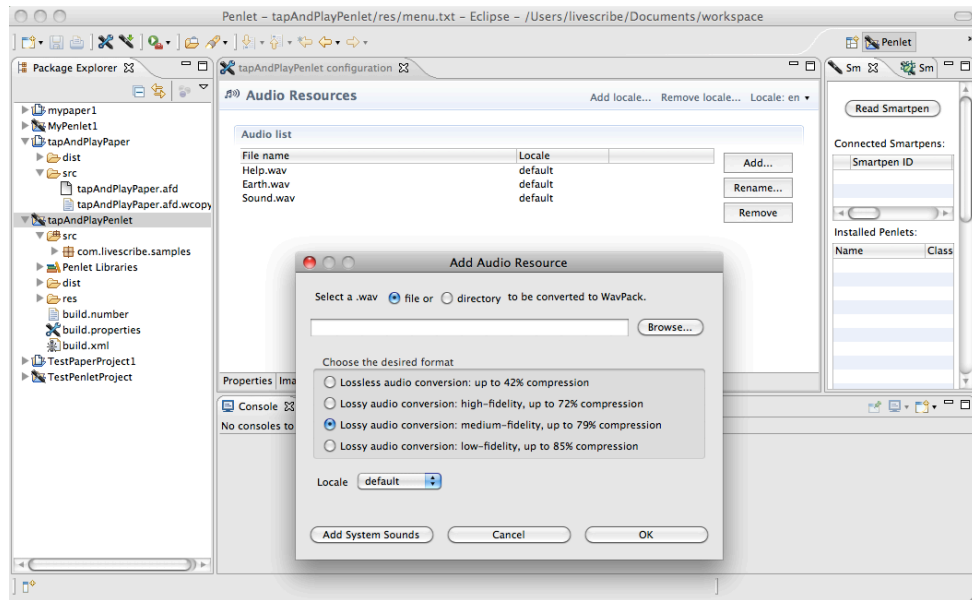
Getting Started with the Livescribe Platform SDK



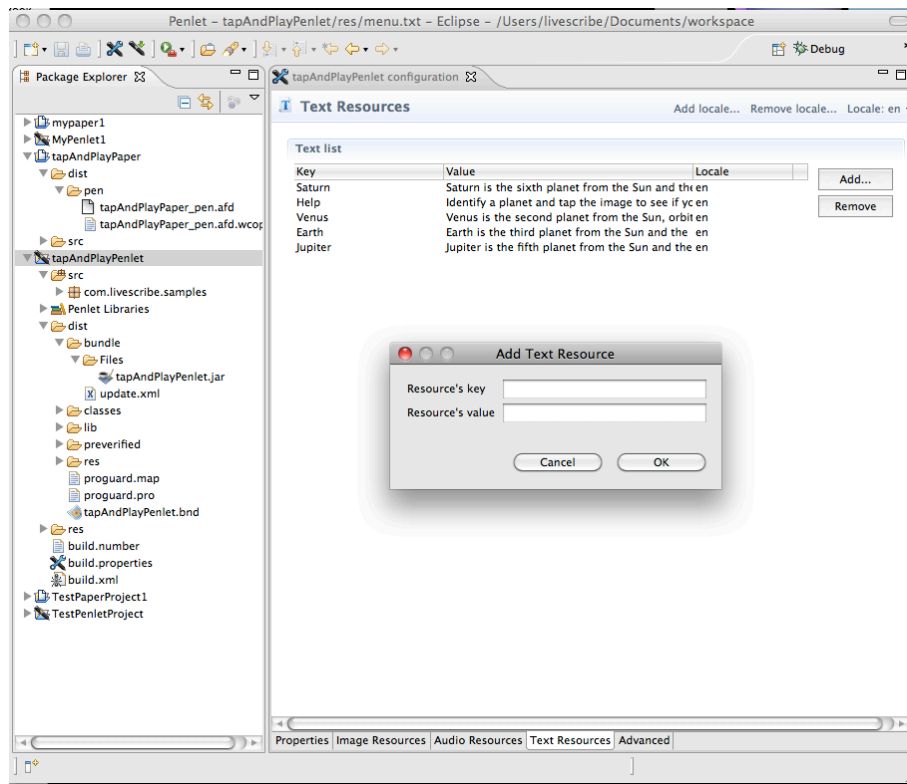
To modify the sample Tap-and-Play application:

1. Import the Tap and Play penlet and paper projects into Eclipse. Choose **File > Import > General > Existing Projects into Workspace**.
2. Browse to the **Samples** folder in the SDK installation folder.
3. Select the **TapAndPlay** folder and click **OK**. This imports both the Paper and Penlet projects into your Eclipse IDE.
4. Edit the active area names on the paper to match target resource names you want to associate with the active areas. For example, to create a Name the Animal application, you could change active area name "Earth" to "Cat".
5. Add a corresponding Cat audio file so it will play automatically when the user taps on the Cat active area. The easiest way to add an audio file is to use the Penlet Configuration Editor and go to the Audio Resources tab. You can add and remove audio resources from there. Click **Add** to launch the Audio Resource converter and locate your audio file.

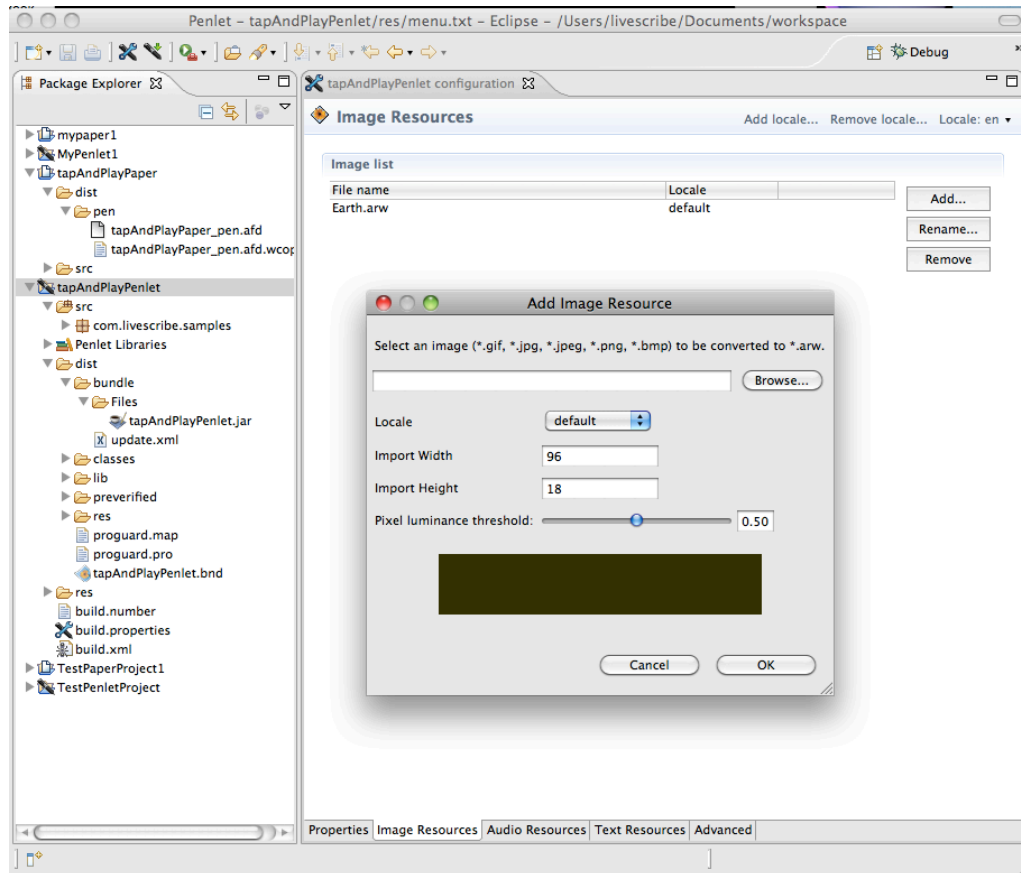
Getting Started with the Livescribe Platform SDK



6. If you want to display text when the “Cat” area is tapped, add a text resource named “Cat”. The quickest way to edit the messages file is to use the **Penlet Configuration Editor** and go to the Text Resources tab. You can add and remove text resources from there.



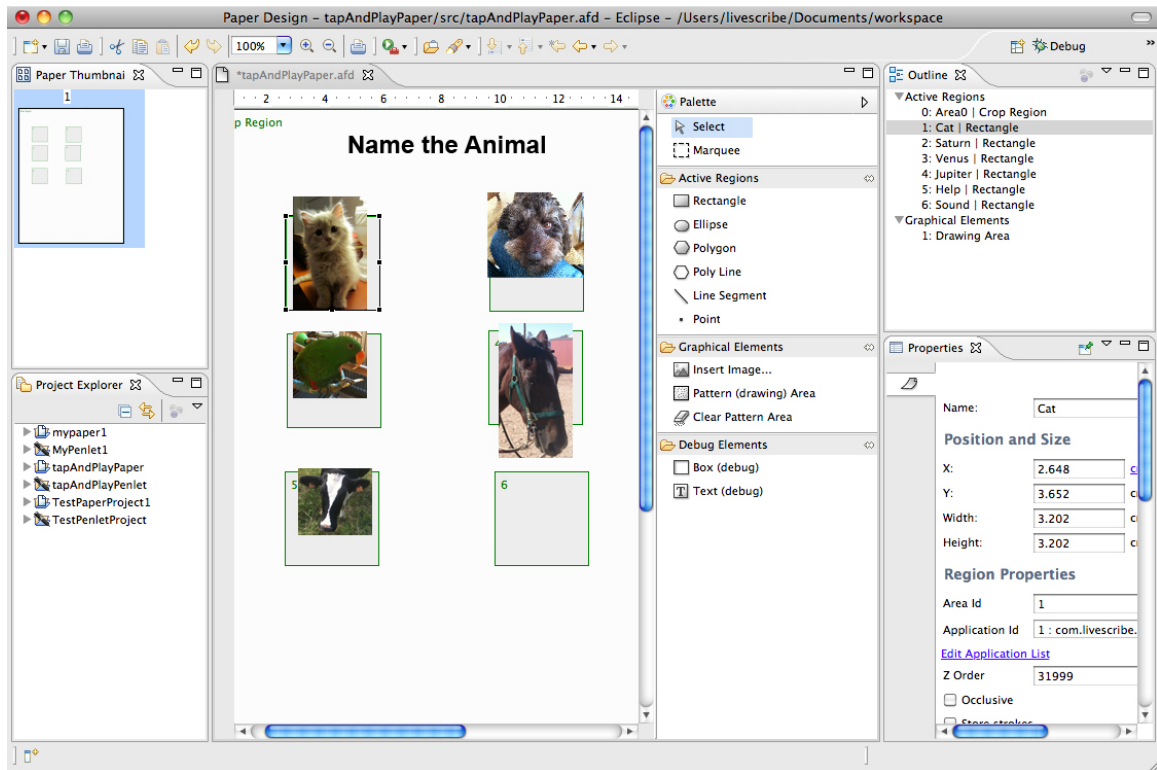
7. If you want to show an image instead of a text string on the display, create an ARW image using the provided ARW converter. Give the file the same name as the active area. The best way to do this is to use the **Penlet Configuration Editor** and go to the Image Resources tab. Click Add to launch the ARW converter. Locate the image file and set its properties.



8. Using the Paper Designer, optionally change the imported background to show the images you want the end user to tap on. For example, for the Name the Animal application, you could replace the Name the Planet background image with one of animal images. You can use the current active areas on the sample paper, or optionally add or change the active areas for your Tap and Play paper product.

Note: The SDK requires images in EPS format. For details on how to import images and use the Paper Designer to create and modify active areas, refer to the *Developing Paper Products* guide.

Getting Started with the Livescribe Platform SDK



9. When you have finished editing the AFD file and updated the penlet resources, deploy the penlet to your smartpen by right-clicking the Penlet project and selecting **Deploy Penlet**.
10. Print out the paper product and test the application. See **Print and Test the Paper Product and Penlet** for help with printing.
11. From the Project Explorer, select the source AFD file in the src folder, and right-click to select **Deploy to Smartpen**.

Note: If you are using Windows, you can deploy the penlet to the Livescribe Smartpen Emulator and test your penlet and paper products on the desktop. From the emulator, choose **Open Paper Product** and browse to your workspace. Locate the AFD file in the dist folder. Next, use your cursor to tap on the active areas to test your application. You can download and install the Livescribe Smartpen Emulator from the Livescribe Developer site, and read the *Livescribe Smartpen Emulator User Guide* for details.

Programming a Hello World Penlet

This section walks through the creation of a very simple penlet, called **HelloWorld**. This penlet will appear as one of the available penlets on your smartpen's Application menu. When launched, it will display the text 'Hello World' and play an audio file that says 'Hello World'. To launch the penlet, you need a Nav Plus. No other paper is required.

Using the Livescribe Penlet Project Creation Wizard

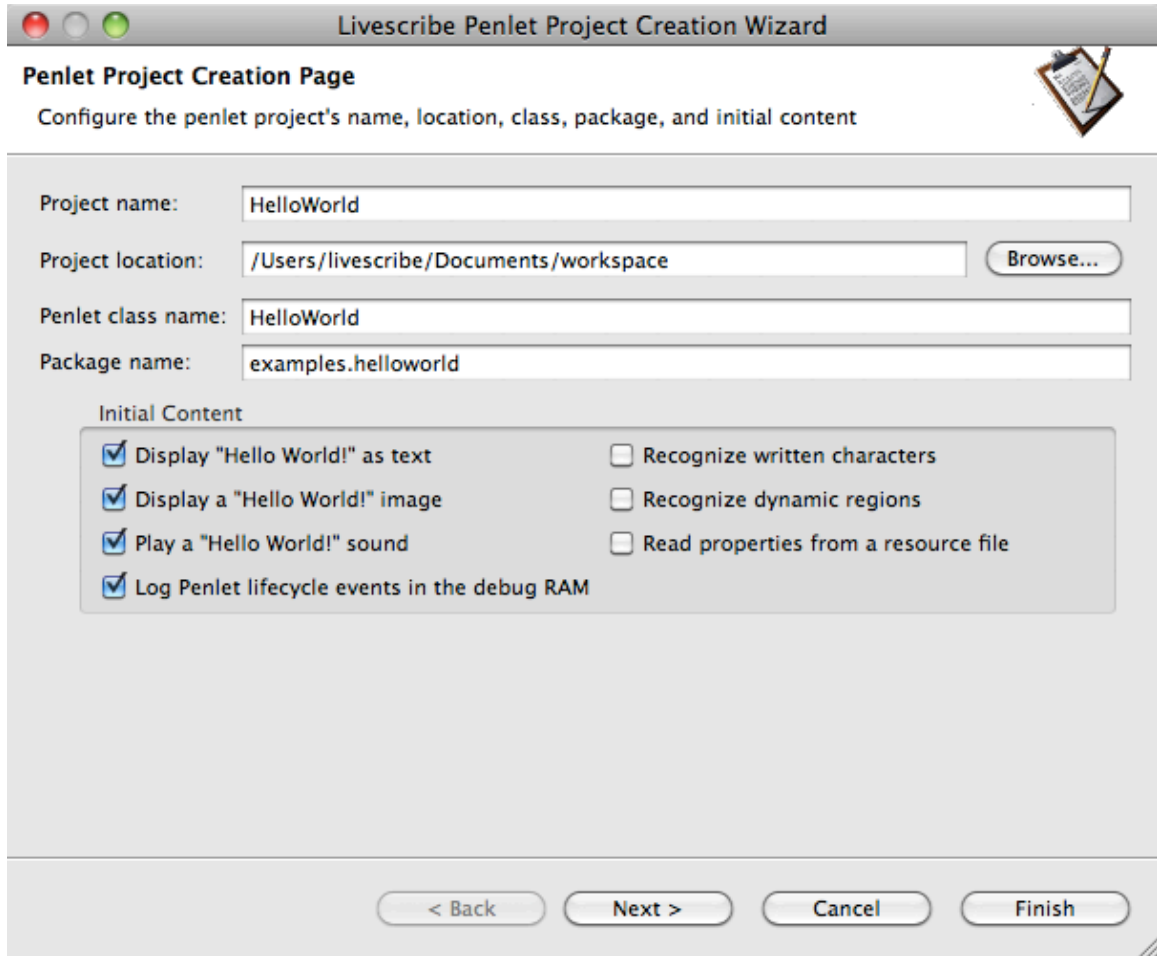
The numbered steps below are actions you take to create the sample. We will use the Livescribe Penlet Project Creation Wizard to create the project. Then we will deploy (install) the penlet to your smartpen, or alternatively to the Livescribe Smartpen Emulator. In between steps, we will stop to explore the dialogs and wizards of the Livescribe Penlet feature and examine some of the source code generated by the wizard.

1. Select **Window > Open Perspective > Penlet**. The Penlet perspective should appear, with the views and toolbars you need already displayed.
2. Select **File > New > Project...** and open the **Livescribe** entry in the New project dialog. Select **Livescribe Penlet Project** and click **Next**.

The project wizard constructs a complete file hierarchy for the penlet, including directories for source files, class files, and resources. It creates an Ant script and adds libraries required for penlet development.

The first page of the Livescribe Penlet Project Creation wizard is the Penlet Project Creation Page, which consists of two sections.

The top half asks for the project name, the project's folder, the penlet class name, and the package name. All four fields are required. The usual rules apply to Java class and package names: no reserved words and no unusual punctuation.



The screenshot shows a macOS-style window titled "Livescribe Penlet Project Creation Wizard". Inside, the "Penlet Project Creation Page" is displayed. Below the title bar, a subtitle reads "Configure the penlet project's name, location, class, package, and initial content". The form contains several input fields: "Project name:" with the value "HelloWorld", "Project location:" with the value "/Users/livescribe/Documents/workspace" and a "Browse..." button, "Penlet class name:" with the value "HelloWorld", and "Package name:" with the value "examples.helloworld". Below these fields is a section titled "Initial Content" containing two columns of checkboxes. The first column has four checked options: "Display 'Hello World!' as text", "Display a 'Hello World!' image", "Play a 'Hello World!' sound", and "Log Penlet lifecycle events in the debug RAM". The second column has three unchecked options: "Recognize written characters", "Recognize dynamic regions", and "Read properties from a resource file". At the bottom of the window are four buttons: "< Back", "Next >", "Cancel", and "Finish".

The bottom half allows you to initialize the source file in your penlet project.

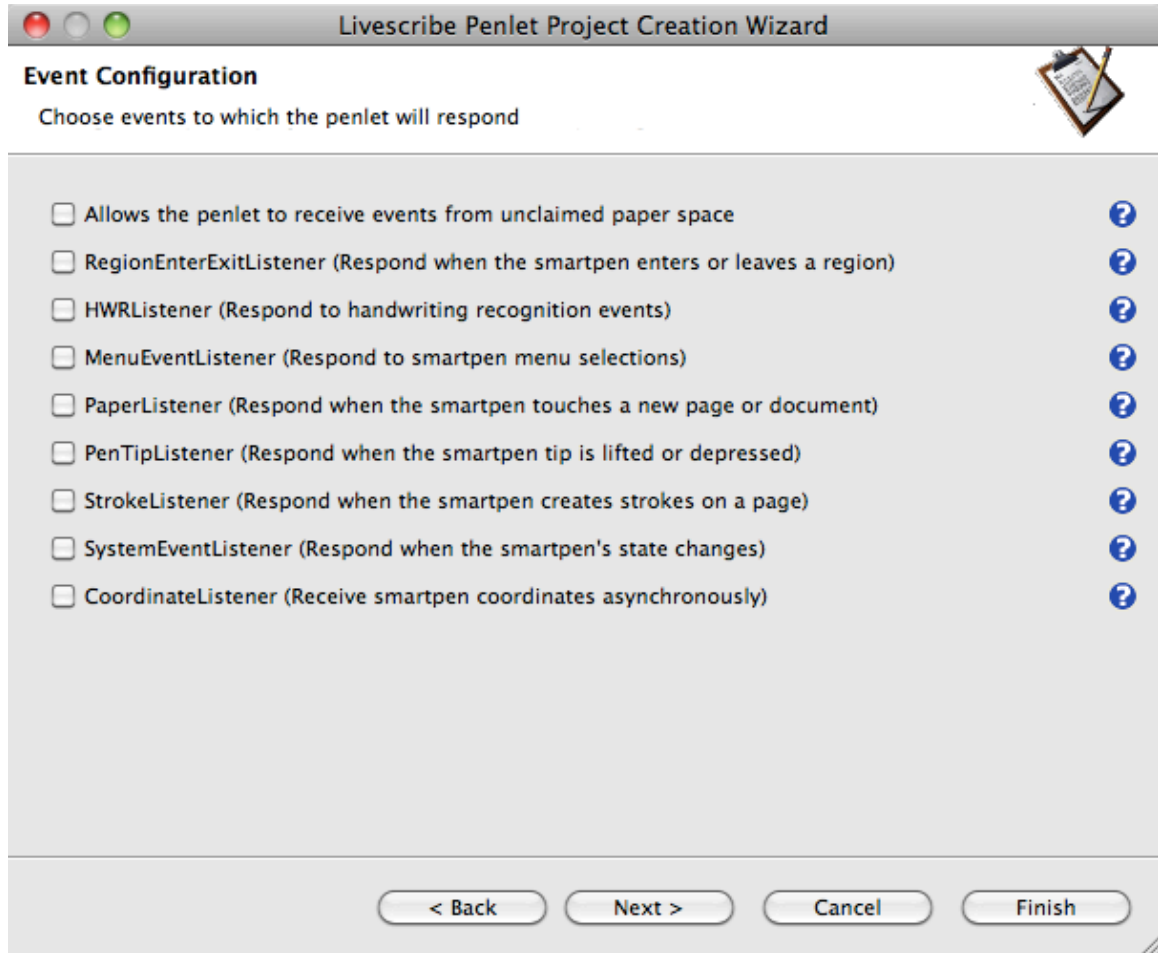
Several different project templates are available. Each one will cause the appropriate source code and resources to be included in your new project.

You can create several projects, selecting different options each time. In this way, you will quickly become familiar with the UI of the Penlet feature and the source code required for common smartpen functionality.

3. In the Penlet Project Creation Page, enter HelloWorld as the name of both the project and the penlet class.
Enter examples.helloworld as the name of the package.
4. Choose these options in the Initial Content group:
 - **Display "Hello World!" as text.**
 - **Display a "Hello World!" image.**
 - **Play a "Hello World!" sound.**
 - **Log Penlet lifecycle events in the debug RAM.**

5. Click **Next** at the bottom of the page.

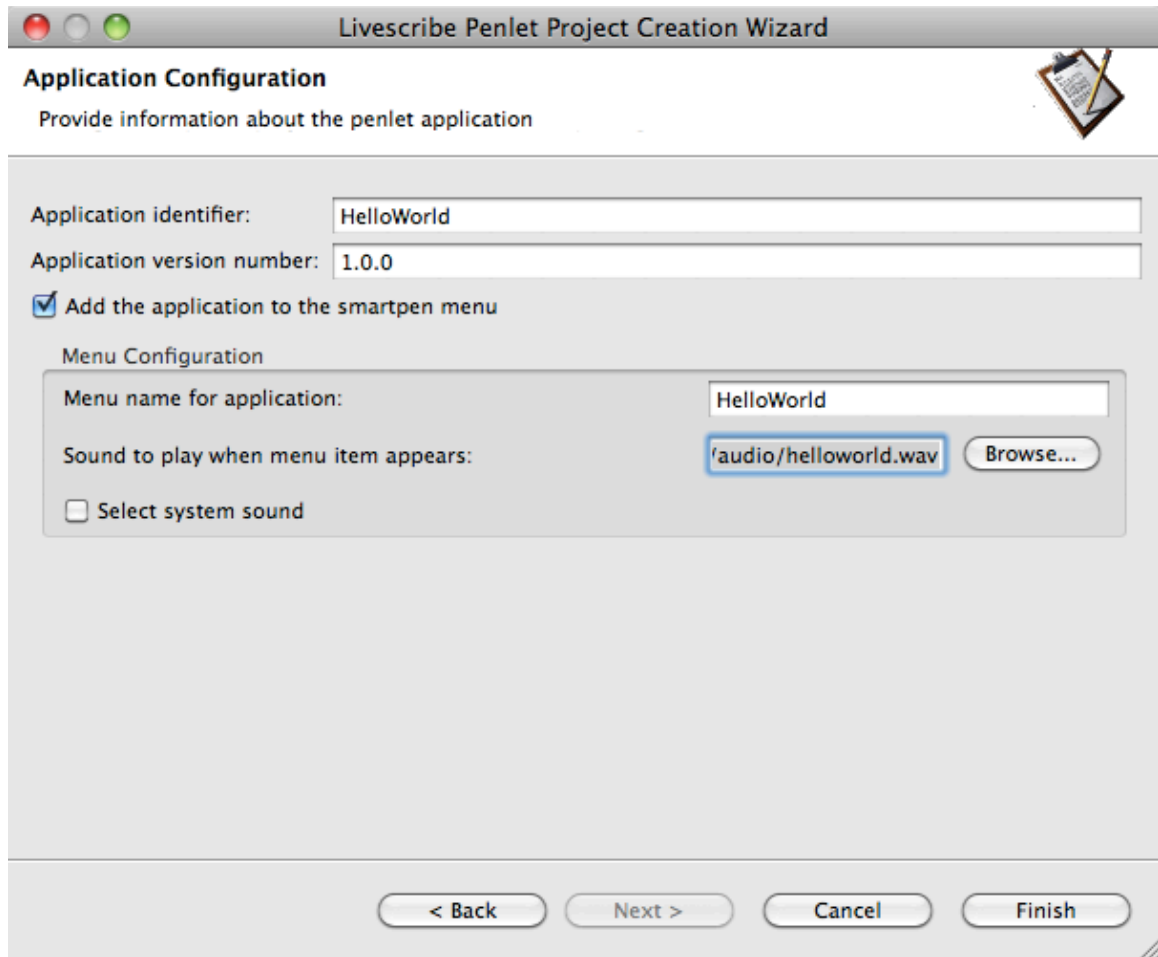
The Event Configuration page appears. On this page you can request that one or more event listeners be added to your project. An event listener is a Java interface that your penlet's primary class must implement. For each event listener you select, the code generator in the wizard creates stubs of the event handlers that make up the interface. You are responsible for providing the actual code of the event handlers.



In this walkthrough, we will not make any selections on this page. The selections we made on the Penlet Project Creation Page will add the appropriate event handlers *and* the code needed to implement them. Examining this generated code can be very instructive. You may wish to create several penlet projects, experimenting with different options on the Penlet Project Creation page of the wizard. With that experience, you will be ready to run the wizard and select options on the Event Configuration page.

6. In the Event Configuration page of the wizard, click **Next**.

The Application Configuration page appears.



The screenshot shows a macOS-style window titled "Livescribe Penlet Project Creation Wizard". The main heading is "Application Configuration" with a subtext "Provide information about the penlet application". There is a small icon of a notepad and pencil in the top right corner. The form contains the following fields and options:

- Application identifier:** A text field containing "HelloWorld".
- Application version number:** A text field containing "1.0.0".
- Add the application to the smartpen menu:** A checked checkbox.
- Menu Configuration:** A sub-section containing:
 - Menu name for application:** A text field containing "HelloWorld".
 - Sound to play when menu item appears:** A text field containing "'audio/helloworld.wav'" and a "Browse..." button.
 - Select system sound:** An unchecked checkbox.

At the bottom of the window are four buttons: "< Back", "Next >", "Cancel", and "Finish".

On this page you can modify:

- Name of the built penlet. This name is used in the Smartpen Info view.
- Version number of the penlet
- Whether the penlet appears on the smartpen's main menu. If it does, the name that appears in the smartpen's main menu for the penlet
- The sound played when the penlet's name rolls into view in the smartpen's main menu. You can play the default system sound or select a custom audio file.

The default values are as follows:

- Built penlet has the same name as the project
- Penlet version is 1.0.0.
- Penlet will appear as an item on the smartpen's main menu

Getting Started with the Livescribe Platform SDK

- Penlet's name on the main menu is the same as the project name
- No audio file plays when the penlet's name rolls into view in the smartpen's Applications menu.

Note: The **Sound to play when menu item appears** audio file is played only when the user is scrolling through the smartpen's Applications menu. When the penlet's name rolls into view in the OLED, the sound plays. If you prefer the default system sound instead, check the **Select system sound** checkbox.

In our Very Simple Penlet sample, the code plays a `HelloWorld.wav` sound when the penlet is *launched*. (We obtained this behavior by selecting the "Play a Hello World! sound" checkbox on the Penlet Creation Page of the Livescribe Penlet Project Creation wizard.) To launch a penlet, the user selects the penlet's name when it appears in the main menu.

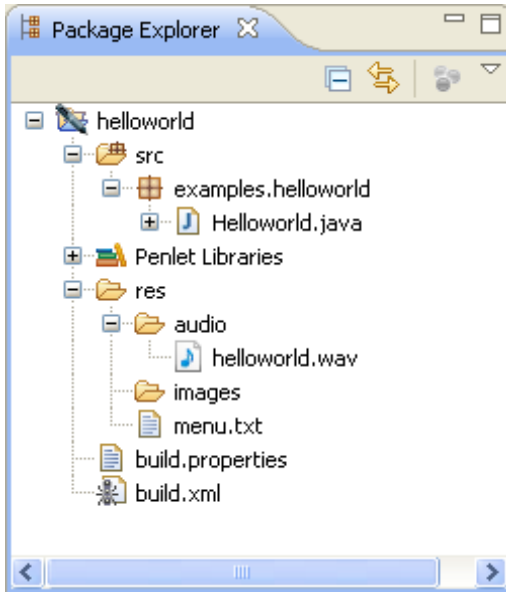
7. On the Application Configuration page of the wizard, do the following:
 - Make sure that the **Add the application to the smartpen menu** checkbox is selected.
 - In the **Menu** name for application field, change `HelloWorld` to `Hello World Sample` (which is a friendlier name to display on the smartpen's main menu).

The *wizard* writes this information to `menu.txt`, and places the file in the project's `res` directory.

8. On the Application Configuration page of the wizard, Click **Finish** and wait for the wizard to complete your project. This can take a few seconds.

The Livescribe Penlet Project Creation wizard creates a `HelloWorld` penlet project *with* resources and source code that matches the selections you made. The Penlet perspective should be active, with all the views and actions necessary to code and deploy a penlet.

The wizard has made the following changes: First, the text editor now displays the source code in the `HelloWorld.java` source file. Second, a new project appears in the Eclipse Package Explorer called `HelloWorld`. If you open its folders, you'll see a structure similar to the following:



If you examine the code in `HelloWorld.java`, you'll see that the following code has been inserted into `activateApp()`:

```
public void activateApp(int reason, Object[] args) {  
    this.logger.info("activated.");  
  
    if (reason == Penlet.ACTIVATED_BY_MENU) {  
        ImageResource helloworldImage =  
            context.getResourceBundle().getImageResource("helloworld");  
  
        this.label.draw(context.getResourceBundle().getTextResource("helloWorld  
.text"),  
            helloworldImage.getImage(), true);  
  
        this.display.setCurrent(this.label);  
  
        SoundResource helloworldSound =  
            context.getResourceBundle().getSoundResource("helloworld");  
  
        this.player.play(helloworldSound);  
    }  
}
```

This first line checks whether the penlet has been activated through menu selection.

If so, the second and third lines tell the smartpen to display "Hello World" on the OLED. The fourth line tells the smartpen to play the `helloworld.wav` audio file. In the Project Explorer, drill down in the `HelloWorld` project by clicking the plus icon

next to a folder. The audio file should appear in the HelloWorld/res/audio directory.

Now that you've created your HelloWorld penlet project, it's time to package it and install it on the smartpen.

9. Dock your smartpen in its cradle and plug the USB connector in to your computer.
10. Click the **Smartpen Info** tab, if necessary, to display the Smartpen Info view. Click on the **Read Smartpen** button.

A progress dialog should appear that says: Collecting smartpen information.

Identifying information about your smartpen should appear in the table below **Connected Smartpens**. The **Mode** value should be Normal. You will probably also see data in the Installed Penlets and Installed Documents tables.

11. If you have more than one smartpen connected, make sure the checkbox of the correct smartpen is selected in the **Connected Smartpens** table.
12. In the Project Explorer, right-click on the project name, which is the top-most node in each project. In our case, it is the text **HelloWorld** with the penlet project icon in front of it:  HelloWorld
13. Select **Deploy Penlet** from the popup menu.

Important: Wait until all progress messages in Eclipse and on the smartpen itself have disappeared. Your smartpen should be displaying the date and time before you proceed.

Note: If you prefer, you can deploy the penlet to the Livescribe Smartpen Emulator and test your penlet and paper products on the desktop. The emulator functions like a physical smartpen. For more information, download and install the Livescribe Smartpen Emulator from the Livescribe Developer site. Read the *Livescribe Smartpen Emulator User Guide* to learn how to install penlets and paper products to it.

Your project is pre-verified, packaged into a JAR, and deployed to your smartpen.

You should check that the deployment was successful. First, consult the Smartpen Info view in Eclipse to see if your HelloWorld sample appears in the Installed Penlets table.

14. In the **Installed Penlets** table of the Smartpen Info view, find your penlet. You may need to scroll.

15. On your Livescribe smartpen, try actually running the HelloWorld penlet.

1. Remove the smartpen from its cradle
2. Tap down on the directional arrow of any Nav Plus and scroll through the smartpen's main menu until you reach the Applications item.
3. Tap right.
4. You are now in a submenu that contains the names of all penlets that you installed on the smartpen.
5. Tap down to scroll through your applications. (There may be only one.)
6. Tap right when you see "Hello World Sample" to activate your penlet.

This activation plays the `helloworld.wav` audio file, which welcomes you with "Hello World!"

Programming a Simple Hello Open Paper Penlet

In this example, we create a different simple penlet, called **HelloOpenPaper**. We assume that you have already created the previous sample, **HelloWorld**, and are familiar with the basics of the **Livescribe Penlet Project Creation Wizard**. If not, please go to Programming a Hello World Penlet

The **HelloOpenPaper** penlet makes use of Livescribe Open Paper. When launched from your smartpen's Application menu, the penlet:

1. Displays the text 'Draw a shape' on the smartpen OLED.
2. The user draws a polygon of some kind on Open Paper. In response, the penlet creates a rectangular dynamic region around that polygon.
3. The penlet then displays text describing the rectangle (or "box"). The text on the smartpen's OLED reads: "Created new box
ID:XX,left:YY,top:ZZ,height:HH,width:WW".

This behavior of the penlet demonstrates that it has "seen" the polygon that the reader created on Open Paper and created a dynamic region around it.

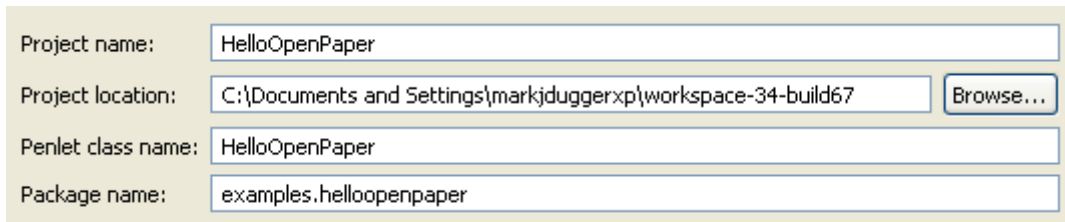
Getting Started with the Livescribe Platform SDK

The numbered steps below are actions you take to create the sample. We will use the **Penlet Project Creation Wizard** to create the project. Then, we will deploy (install) the penlet to your smartpen.

1. Select **Window > Open Perspective > Penlet**. The Penlet perspective should appear, with the views and toolbars you need already displayed.
2. Select **File > New > Project...** and open the **Livescribe** entry in the New project dialog. Select **Livescribe Penlet Project** and click **Next**.

The Livescribe Penlet Project Creation Wizard starts. It constructs a complete file hierarchy for the penlet, Ant scripts, and libraries required for penlet development.

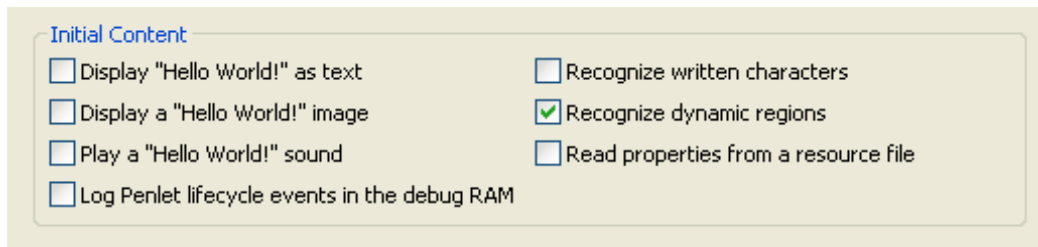
The Project Creation Page is divided into two sections. The top section asks for the project name, the project's directory location, the penlet class name, and the package name.



The screenshot shows a form with four input fields. The first field is labeled 'Project name:' and contains the text 'HelloOpenPaper'. The second field is labeled 'Project location:' and contains the text 'C:\Documents and Settings\markjduggerxp\workspace-34-build67', with a 'Browse...' button to its right. The third field is labeled 'Penlet class name:' and contains the text 'HelloOpenPaper'. The fourth field is labeled 'Package name:' and contains the text 'examples.helloopenpaper'.

All four fields are required. The usual rules apply to Java class and package names: no reserved words and no unusual punctuation.

The bottom half of the Project Creation Page allows you to initialize the source file in your penlet project. We want a penlet that can recognize dynamic regions drawn by the user on Open Paper.



The screenshot shows a section titled 'Initial Content' with a list of five options, each with a checkbox. The first four options are: 'Display "Hello World!" as text', 'Display a "Hello World!" image', 'Play a "Hello World!" sound', and 'Log Penlet lifecycle events in the debug RAM'. The fifth option is 'Recognize written characters'. The sixth option is 'Recognize dynamic regions', which has a checked checkbox. The seventh option is 'Read properties from a resource file'.

3. In the Penlet Project Creation Page, enter HelloOpenPaper as the name of both the project and the penlet class.
Enter examples.helloopenpaper as the name of the package.
4. Choose one option in the Initial Content group:
 - Recognize dynamic regions

5. Click **Next** at the bottom of the page.

The Event Configuration page appears. On this page you can request that one or more event listeners be added to your project. For each event listener you select, the wizard creates stubs of the event handlers that make up the listener interface. You must provide the actual code of these event handlers.

In this walkthrough, we will not make any selections on this page. The selections we made on the first page will add the appropriate event handlers *and* the code needed to implement them. Examining this generated code can be very instructive.

6. In the Event Configuration page of the wizard, click **Next**.

The Application Configuration page appears.

7. On the Application Configuration page of the wizard, do the following:

- Make sure that the **Add the application to the smartpen menu** checkbox is selected.
- In the Menu name for application field, change `HelloOpenPaper` to `Hello Open Paper` (which is a friendlier name to have on the smartpen's main menu).

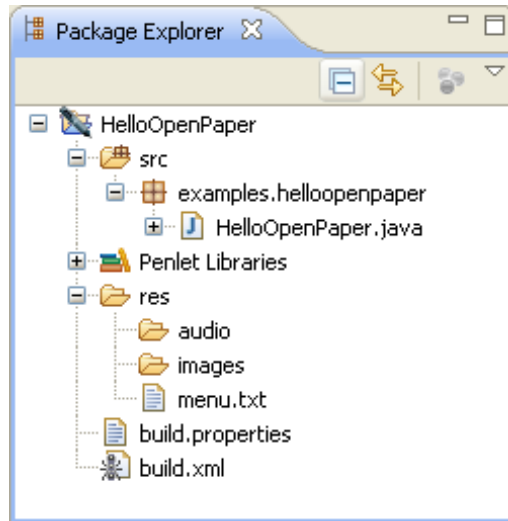
The wizard writes this information to `menu.txt`, and places the file in the project's `res` directory.

8. On the Application Configuration page of the wizard, Click **Finish** and wait for the wizard to complete your project. This can take a few seconds.

The Livescribe Penlet Project Creation wizard creates a `HelloOpenPaper` penlet project with source code that matches the selections you made on first page. The Penlet perspective should be active, with all the views and actions necessary to code and deploy a penlet.

The Eclipse window changes in two ways. First, the text editor displays the source code in the `HelloOpenPaper.java` source file. Second, a new project appears in the Eclipse Package Explorer called `HelloOpenPaper`. If you open its folders, you'll see a structure similar to the following:

Getting Started with the Livescribe Platform SDK



If you examine the code in `HelloOpenPaper.java`, you will see that the automatic code generator in the wizard has inserted the following code into

`activateApp()`:

```
public void activateApp(int reason, Object[] args) {

    this.display.setCurrent(label);
    this.context.addStrokeListener(this);
    this.context.addPenTipListener(this);
    this.context.addRegionEnterExitListener(this);
    this.label.draw(

this.context.getResourceBundle().getTextResource("dyn.instruction.text"
),
        true);
    context.addPenTipListener(this);
    context.addRegionEnterExitListener(this);
}
```

Check the signature of the `HelloOpenPaper` class and you will see that this class implements the `StrokeListener` interface. The third line in the code snippet above attaches the `HelloOpenPaper` class to the penlet context.

That is enough code for the moment. After we build and deploy this sample penlet, we shall return to the source code, examining the code that recognizes dynamic regions drawn by the penlet user. For now, let's continue.

Now it's time to build your new penlet and deploy (install) it on the pen.

9. Dock your smartpen in its cradle and plug the USB connector in to your computer.
10. Click the Smartpen Info tab, if necessary, to display the Smartpen Info view. Click on the **Read Smartpen** button.

A progress dialog should appear that says: Collecting smartpen information.

Identifying information about your smartpen should appear in the table below

Connected Smartpens. The **Mode** value should be Normal. You will probably also see data in the Installed Penlets and Installed Documents tables.

11. If you have more than one smartpen connected, make sure the checkbox of the correct smartpen is selected in the **Connected Smartpens** table.

12. In the Project Explorer, right-click on the project name, which is the top-most node in each project. In our case, it is the text **HelloOpenPaper** with the penlet project icon in front of it:



13. Select **Deploy Penlet** from the popup menu.

IMPORTANT: Wait until all progress messages in Eclipse and on the smartpen itself have disappeared. Your smartpen should be displaying the date and time before you proceed.

Your project is pre-verified, packaged into a JAR, and deployed to your smartpen.

You should check that the deployment was successful. First, consult the Smartpen Info view in Eclipse to see if your HelloOpenPaper sample appears in the Installed Penlets table.

14. Click the **Smartpen Info** tab, if necessary, to display the Smartpen Info view. Click on the **Read Smartpen** button.

A progress dialog should appear that says: Collecting smartpen information.

15. In the **Installed Penlets** table, find your penlet. You may need to scroll. The name will be HelloOpenPaper.

16. On your smartpen, try actually running the HelloOpenPaper penlet.

- Remove the Livescribe smartpen from its cradle
- Tap down on the directional arrow of any Nav Plus and scroll through the smartpen's main menu until you reach the Applications item.
- Tap right.
- Tap down to scroll through the submenu, if necessary.
- Tap right to activate your HelloOpenPaper penlet.

This activation displays text that says "Draw a Shape." The penlet is waiting for you to draw a single-stroke shape on Open Paper.

17. Find some Open Paper, such as the blank, middle portion of a Livescribe notebook.
18. Draw a rectangle on the paper—all in *one* stroke. In other words, don't lift your smartpen for each side.
19. Quickly look at the OLED.
It should display the height and width of the rectangle you just drew.
20. Draw another single-stroke polygon on the paper and look at the display on the OLED.

You have successfully created the HelloOpenPaper sample penlet!

The strokeCreated Method

We will now take a closer look at some of the code for the HelloOpenPaper penlet.

1. In Eclipse, find the text editor displaying the source code for HelloOpenPaper class.
2. Scroll down, if necessary, until you see the strokeCreated method.

The `strokeCreated` method is an event handler specified by the `StrokeListener` interface. Since the `HelloOpenPaper.java` class implements that interface, the Livescribe Penlet Project Creation Wizard inserted code in the `strokeCreated` method for you.

```
public void strokeCreated(long time, Region regionId, PageInstance
page) {

    this.logger.info("strokeCreated(regionId = " +
regionId.getIdString() + ")");

    // If this is a stroke in a region we already created, we'll skip
it.
    // If this is a stroke in an area we haven't seen before it will
have an areaId of 0.
    if (regionId.getAreaId() > 0) return;

    // Get the strokes for the page on which this new stroke was
written
    StrokeStorage strokeStorage = new StrokeStorage(page);

    // Get the stroke that was just created
    Stroke stroke = strokeStorage.getStroke(time);
```

Getting Started with the Livescribe Platform SDK

```
Rectangle boundingBox = stroke.getBoundingBox();

// Get the regions currently on this page
RegionCollection regions =
this.context.getCurrentRegionCollection();

// Create a Region and add it to the document on the pen
// Note: We always use the same area id because the function
// each region is to perform (telling us where it is on the paper)
is the same.
// The regions will still have unique IDs however because
// the Z-Order is always unique.

Region newRegion = new Region(AREA_ID, false);

regions.addRegion(boundingBox, newRegion);

String boxString = getDisplayString(newRegion, boundingBox);

this.label.draw(

this.context.getResourceBundle().getTextResource("newBox.label") +
boxString,
    true);

    this.logger.info("strokeCreated:  " + boxString);
}
```

The code required for this sample penlet can be summarized as follows:

1. Obtain a Stroke object representing the stroke drawn by the user:
 - Create a `StrokeStorage` object.
 - Call the `getStroke` method on that object to return a `Stroke` object for the stroke that the user created at the `time` passed in to this event handler.
2. Create a shape from that Stroke:
 - a. Call the `getBoundingBox` method on the `Stroke` object. It returns a `Rectangle` that encloses the stroke.
3. Create a region using that shape:
 - a. Create a `Region` object
 - b. Add it to the current `Regions` collection. Notice that when you add the `Region` to the `RegionCollection`, you pass in the rectangle (i.e., `boundingBox`) that you created above.

4. Call the API method `this.label.draw` to display on the smartpen's OLED the left-top corner coordinates and the height and width of that rectangle.

Note that the `getDisplayString()` is a custom method that returns a string describing the rectangle around the user's stroke. The string has the format: "Created new box ID:XX,left:YY,top:ZZ,height:HH,width:WW")

Also note that `getResourceBundle.` and `getTextResource` are API methods for getting text resources. They are stored in a `messages.properties` file and are internationalizable. See "Using Internationalized Text Resources" in *Developing Penlets*.

The `canProcessOpenPaperEvents` Method

Locate the `canProcessOpenPaperEvents` method at the end of the `HelloOpenPaper.java` file. It is another event handler specified by `StrokeListener`. You **must** implement this method and return `true`, if you wish your penlet to be able to receive any events—such as strokes—from Open Paper.

Designing a Simple Paper Project

The easiest way to create a new project for a paper product is to use the Livescribe **Paper Project Creation Wizard**.

The Livescribe Paper Project Creation Wizard

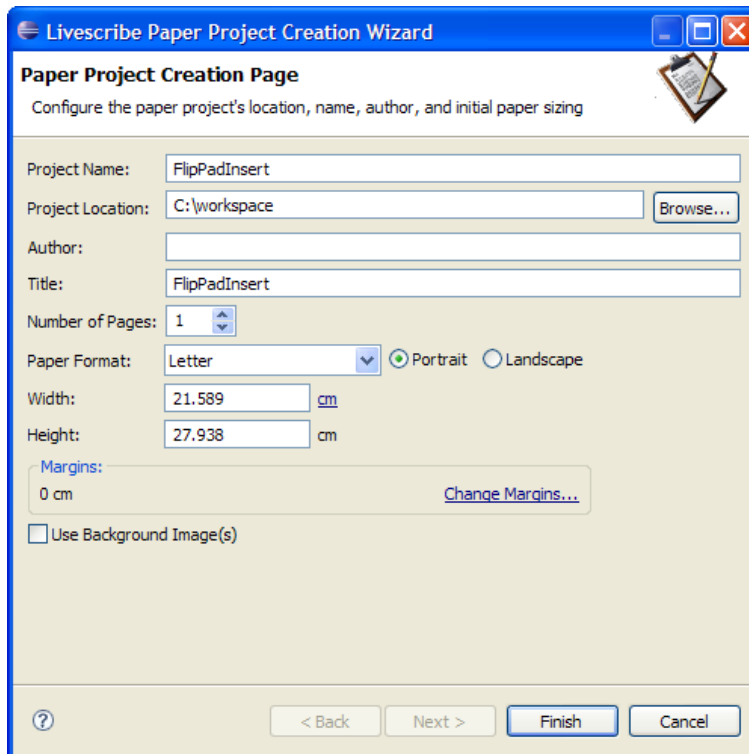
In this example, you will practice creating a paper project using the Livescribe **Paper Project Creation Wizard**. The wizard sets up the project, and creates the electronic version of your paper product known as an AFD.

Livescribe paper products are defined by an AFD file, which contains paper attributes, dot space, images, application linkages, and other relevant information. The wizard will create the main source version of the AFD (under the `src` folder) from which you can print, and a separate AFD file (under the `dist` folder) that will be deployed on the smartpen.

Getting Started with the Livescribe Platform SDK

The purpose of this example is to introduce you to using the Livescribe Paper Project Creation Wizard.

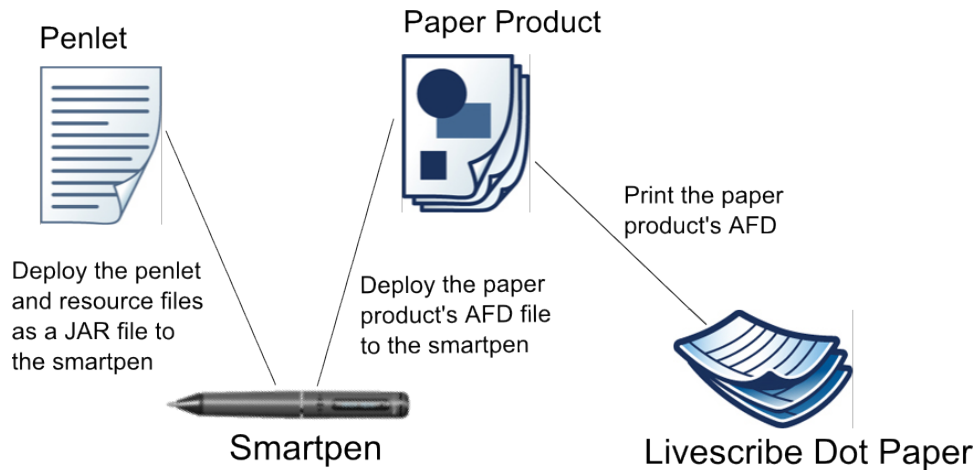
1. Launch Eclipse.
2. Choose **File > New > Livescribe Paper Project**. This starts the Livescribe Paper Project Creation Wizard, and opens up a Paper Product Perspective view in the IDE.
3. Specify properties for the project. Enter the project name, such as **PracticePaperProject** and verify the location of your Eclipse workspace.
4. Uncheck the **Use Background Image(s)** checkbox. You will not need this for this example. (If you had an .eps file to use as a background, you would click **Use Background Image(s)** and import the file on the **Import Image Resources** page.)
5. Leave other parameters as their defaults.



6. Click **Finish**. The wizard will then create and display your paper product and ask if you wish to open the Eclipse perspective (windows and views) for paper design. You can also manually select this perspective from **Window > Open Perspective > Paper Design**.

Programming a Fixed Print Application

This section walks through the basic steps of developing a simple Fixed Print (FP) application. An FP application consists of a paper product and one or more associated (linked) penlets.



For introductory information about Livescribe paper products and penlets, refer to *Introduction to the Livescribe Platform*.

For this example, you will be creating a simple "Tap and Play" application. You will create an Active Region that a smartpen will respond to when the user taps on it. In this case, the response will be to play a "Hello World" audio file.

Note: The quickest way to create a Tap and Play application is to use the sample Tap and Play code provided with the SDK. The sample code is set up so you can quickly edit the paper product areas, their names, and simply add corresponding resource files like audio, ARW images, and text strings to be displayed. To learn more about the sample Tap and Play application, see [A Basic Tap-and-Play Application](#)

The example below walks you through programming a Tap and Play application without the sample Tap and Play code.

Design and Create Page Images

Generally, the first step in creating a Fixed Print application is to design the images for the paper product, including the background and other page art. For this example, however, you will use predefined artwork from Livescribe that is used for the Flip Notepad insert card.

Create a Penlet Project

1. Launch Eclipse.
2. Choose **File > New > Livescribe Penlet Project**. This starts the **Livescribe Penlet Project Creation Wizard**, and opens up a Penlet Perspective view in the IDE.
3. Follow the wizard dialogs to create the basic penlet you want to interact with your paper product. Name the penlet **FlipPadInsertPenlet** with a package name of **com.<yourcompany>.flipPadPenlet**.
4. To help with debugging, check the **Log Penlet lifecycle events in the Debug RAM**. Leave the other checkboxes unchecked. For this application, you will not need them.
5. Click **Next**, and in the next dialog, check the smartpen events you want the penlet to respond to. The wizard will auto-generate code to respond to typical smartpen events. For this example, check the Menu Event Listener and Pen Tip Listener checkboxes.
6. Click **Next**.
7. Provide Application Configuration information, including a specific application identifier and version number. Specify a menu name for the example. Leave blank the Sound to play when the menu is invoked text box. This is not required for this example.
8. Click **Finish**.

Create a Paper Product Project

First, you create the paper product project with the wizard. Then you can use the Paper Designer to complete your design.

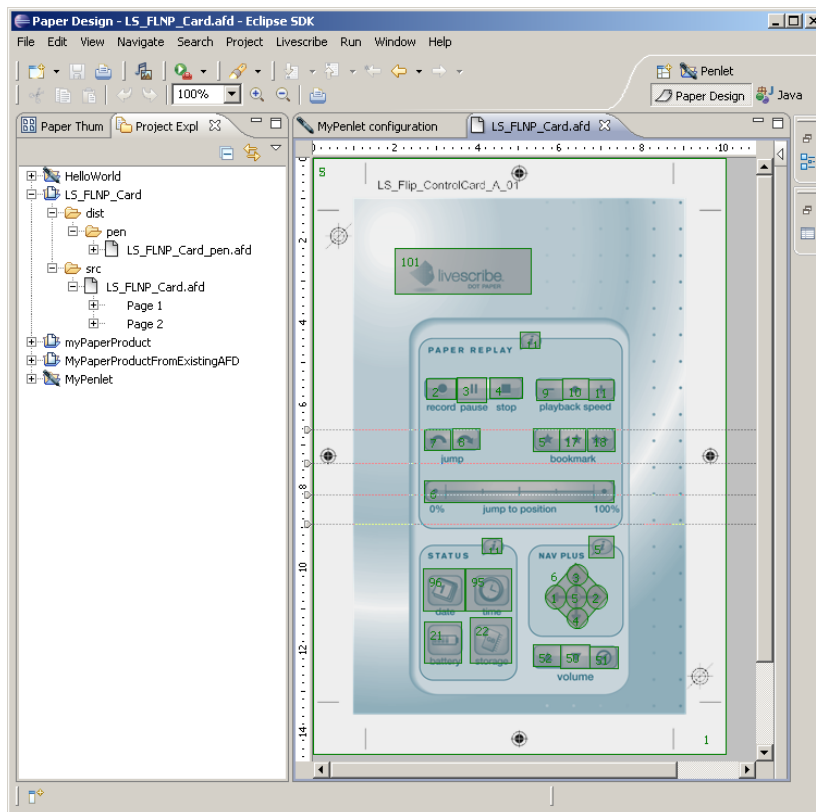
Import the Livescribe Flip Notepad Insert Card AFD File

In this example, you will not be creating an AFD file from scratch. Instead, you will be importing an existing AFD—the Flip Notepad insert card AFD file. This full-featured AFD illustrates several important points that arise when you create and design a paper product for use with your penlet.

1. Choose File >New > Project and select Livescribe Paper Project from Existing AFD in the New Project Wizard.
2. In the import dialog, name the project and browse to its workspace if necessary.
3. Locate the AFD file to import. Browse to the Samples folder in the Livescribe SDK installation package and select the Flip Pad folder and the AFD :

\Samples\FlipPadPaperProject\LS_FLNP_Card.afd

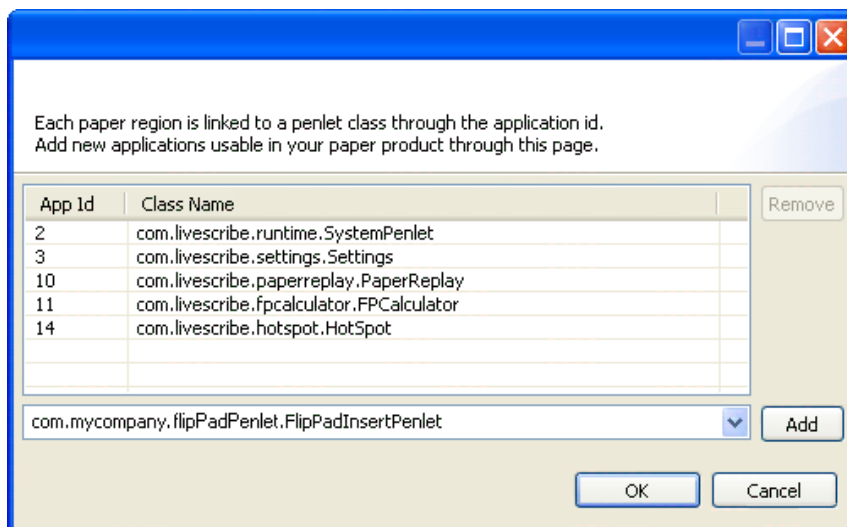
4. Click **Finish** in the Import File system dialog to import the AFD file to your paper product project folder. The imported AFD displays in the Project explorer tree. Double-click **LS_FLNP_Card.afd** to view the AFD in the main drawing area and in the thumbnail views of the Paper Designer.



Link the Paper Product to the Penlet

Associate (or link) the Flip Notepad insert card AFD to the penlet you created earlier (**FlipPadInsertPenlet**). Because you are using an existing Livescribe AFD, notice that the AFD already has standard Livescribe penlets (like Paper Replay) and system penlets linked to it.

1. In Project Explorer, open the folder for your paper product.
2. Open the **src** folder.
3. Double-click the imported AFD file called **LS_FLNP_Card.afd**.
4. Select File > Properties.
5. Select the **Document** tab.
6. Click **Edit Application List** to open the dialog.



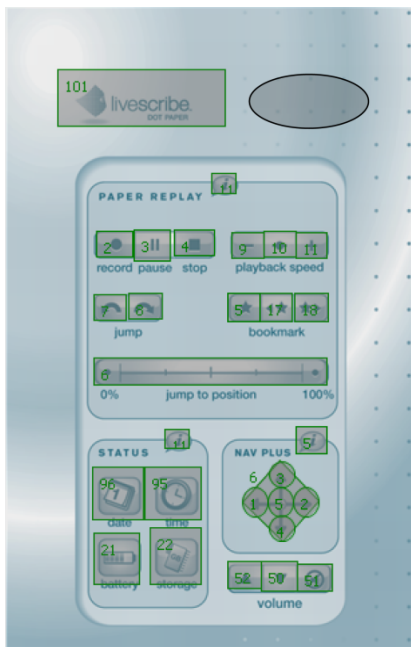
7. From the drop-down list, select the fully-qualified class name for the custom penlet you created earlier. For example:
com.mycompany.flipPadPenlet.flipPadInsertPenlet
where *mycompany* is the name of your company.
8. Click **Add**.
9. Click **OK** to close the Edit Application List.
10. Click **OK** to close the Documents window.

Create Active Regions (Shaping)

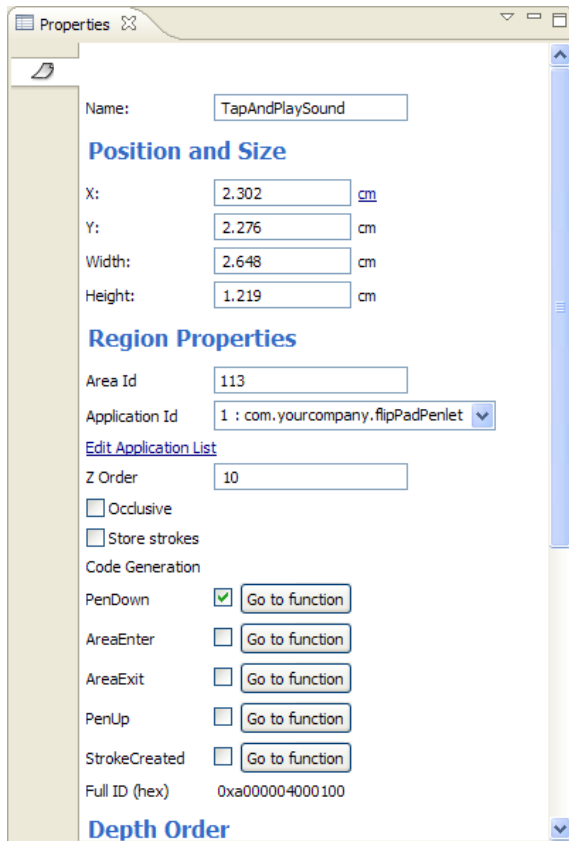
Use the Paper Designer graphical tools to draw active regions (controls, tap-and-play areas, and so on) that your penlet will respond to. This process is also known as *shaping*.

To create an active region on your page:

1. Select a shape (such as an ellipse) from the Active Regions folder in the palette.
2. Click in the editor canvas to place the element on your page. For this example, draw the ellipse next to the Livescribe Dot Paper logo, as shown below.



3. Resize and move the element by clicking and dragging them in the canvas, or edit the properties in the inspector.
4. With the new region selected, use the property inspector to specify the region's name, Area Id, and the penlet that will respond to this region. The Area Id is what links specific regions on the page to the penlet. You can set it to any integer value as long as it is unique for the penlet it is linked to.
For this example, name the region **TapAndPlaySound** and accept the default Area Id.
5. In the Application Id drop down list, select your custom penlet. It should be **1: com.yourcompany.flipPadPenlet**.

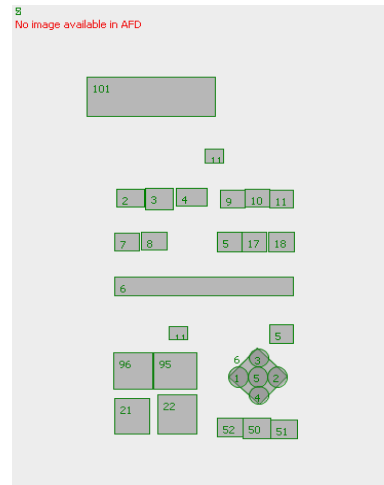
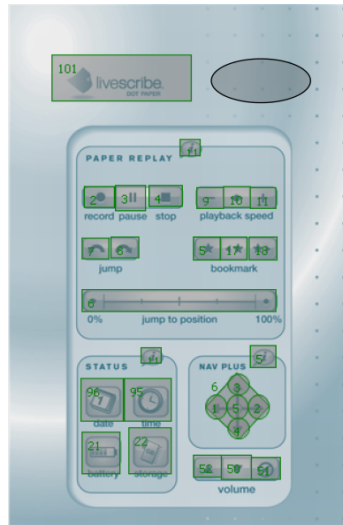


Examine the Livescribe Standard Controls

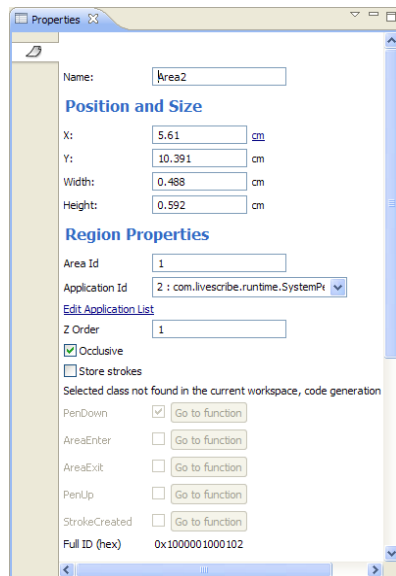
Although you will not be changing them, it is helpful to examine the pre-defined regions on the Livescribe Flip Notepad Insert Card AFD.

1. Examine the Nav Plus control and notice how its Active Regions are drawn (shaped). There are five regions to shape for the center, and the left, right, top, and bottom, and a diamond-shaped "null region". This region is set up to not respond to smartpen events.

Getting Started with the Livescribe Platform SDK



- Also, notice the properties that have been pre-set for one of the regions. For example, the NavPlus left arrow has an Area Id of **1** and the Application Id is **2**: **com.livescribe.runtime.SystemPenlet**.



Auto-generate Event Code for Your Active Region

For each custom Active Region used by your penlet, have the Paper Designer auto-generate event code. This process saves time when programming your penlet. To trigger auto-generation of event code for an Active Region:

1. Select your custom Active Region (the **TapAndPlaySound** region).
2. In the Property Inspector, if not already selected, choose your penlet in the Application Id drop down menu.
(**com.<yourcompany>.flipPadPenlet.flipPadInsertPenlet**).
3. Check the **PenDown** event. Checking this event automatically generates stub code in your penlet for this Active Region.
4. Click **Go to Function** next to PenDown to open the penlet for further editing.

In the java editor, notice that the Paper Designer automatically generated a stub penlet method called **onTapAndPlaySoundPenDown** to respond to a PenDown and other pen tip listener events on this Active Region. You can then use the Penlet editor to define what the behavior should be.

Here is a code listing showing some of the auto-generated code:

```
public void penDown(long time, Region region, PageInstance page) {
    penDownEventDelegator(time, region, page);
}

public void singleTap(long time, int x, int y) {
}

public void doubleTap(long time, int x, int y) {
}

// *** GENERATED METHOD -- DO NOT MODIFY ***
/**
protected boolean penDownEventDelegator(long time, Region region,
PageInstance page) {
    boolean eventHandled = true;
    switch (region.getAreaId()) {
        case 1:
            eventHandled = onTapAndPlaySoundPenDown(time, region,
page);
            break;
        default:
            eventHandled = false;
    }
    return eventHandled;
}
```

```
}  
// *** END OF GENERATED CODE ***  
  
protected boolean onTapAndPlaySoundPenDown(long time, Region  
region,  
    PageInstance page) {  
    return true;  
}  
}
```

Complete the Penlet Programming

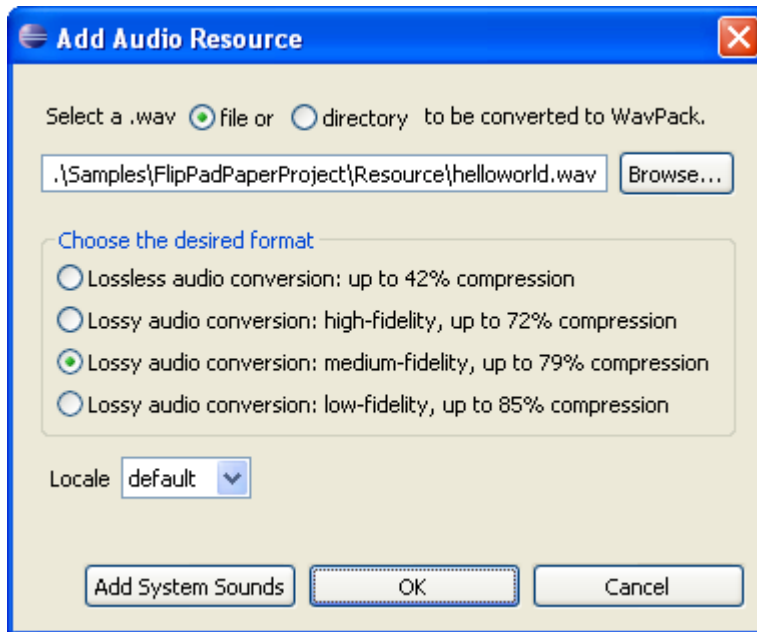
This completes your paper product design. Next, return to the Penlet perspective to complete your penlet code. Refer to the Livescribe SDK Javadoc, *Developing Penlets*, and *User Experience Guidelines* for information about how to design the best penlet for your application goals.

Import the Hello World Audio File

For this example, you will design your application to play a Hello World audio file when the user taps a specific Active Area. To enable this, first import the audio file as a resource:

1. Choose **Window > Open Perspective > Penlet**.
2. Open the **FlipPadInsertPenlet.java** file to view it in the editor.
3. Right click the penlet in the Package Explorer tree and select **Penlet Configuration Editor**
4. Click the **Audio Resources tab** and select **Add**
5. Click **Browse** and Navigate to the Livescribe SDK installation package to select **\Samples\FlipPadPaperProject\Resource\helloworld.wav**

6. Click **OK** to import the file



Set Up the Media Player

To play a sound on a smartpen, your penlet needs to instantiate the MediaPlayer. To do this, add the highlighted lines of code to your penlet. The first line imports the MediaPlayer libraries. The second line imports the SoundResource libraries. The third line declares player as MediaPlayer for this penlet. The fourth line instantiates the MediaPlayer when the penlet is initialized.

```
package com.yourcompany.flipPadPenlet;

import com.livescribe.penlet.Penlet;
import com.livescribe.penlet.Region;
import com.livescribe.ui.MediaPlayer;
import com.livescribe.i18n.SoundResource;
import com.livescribe.afp.PageInstance;
import com.livescribe.event.PenTipListener;

public class FlipPadInsertPenlet extends Penlet implements
PenTipListener {
    private MediaPlayer player;
    public FlipPadInsertPenlet() {
    }
    /* Invoked when the application is initialized. This happens once
    for an application instance.
    */
    public void initApp() {
        this.player = MediaPlayer.newInstance(this);
        this.logger.info("Penlet FlipPadInsertPenlet initialized.");
    }
}
```

```
} }
```

Edit the PenDown Method

Next, edit the `onTapAndPlaySoundPenDown` stub code that was auto-generated by the Paper Designer for your Active Region. Add the highlighted code to instruct the `MediaPlayer` to play the Hello World audio file when the smartpen detects a `penDown` event on the `TapAndPlaySound` Active Region.

```
protected boolean onTapAndPlaySoundPenDown(long time, Region region,
PageInstance page) {

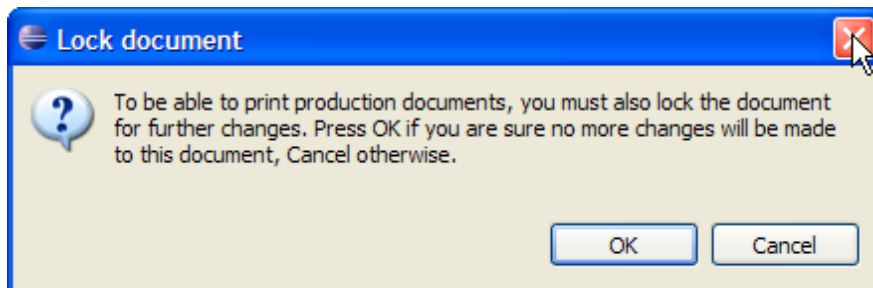
    SoundResource helloworldSound =
        context.getResourceBundle().getSoundResource("helloworld");

    this.player.play(helloworldSound);

    return true;
}
```

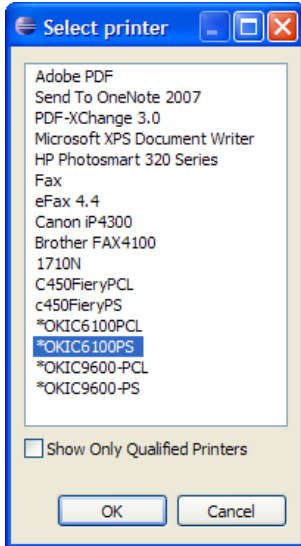
Print and Test the Paper Product and Penlet

1. Deploy the penlet to your smartpen by right-clicking the Penlet project and selecting **Deploy Penlet**.
2. From the Project Explorer, select the source AFD file in the src folder, and right-click to select **Deploy to Smartpen**.
If you prefer, you can deploy the penlet to the Livescribe Smartpen Emulator and test your penlet and paper products on the desktop. Download and install the Livescribe Smartpen Emulator from the Livescribe Developer site, and read the *Livescribe Smartpen Emulator User Guide* for details.
3. Print your paper product using the AFD. Right-click the **LS_FLNP_Card.afd** file in the src folder and choose **Print**.
4. Click **OK** at the **Lock document** prompt.

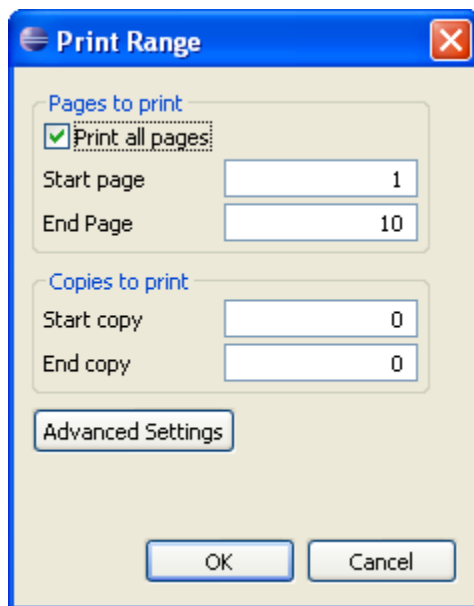


5. In the Select printer dialog, select the printer you want to use and click **OK**.

When printing, you should use a recommended printer. However, other printers can also work. For more information, see the "Recommended Printers" section in *Developing Paper Products*.

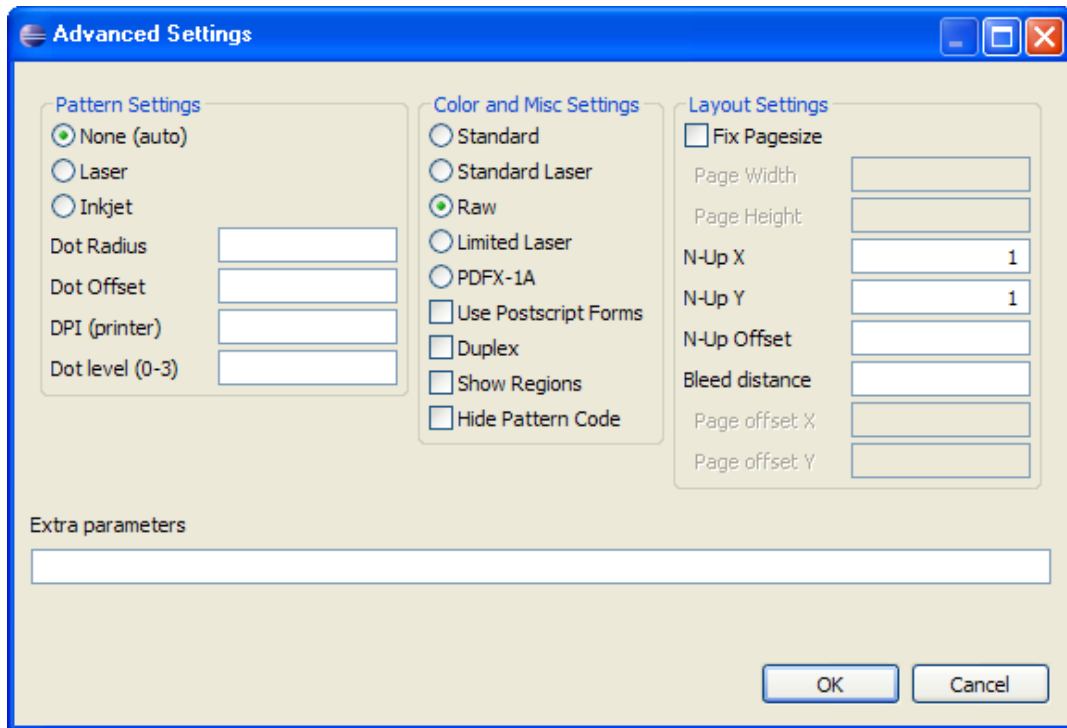


6. In the Print Range dialog, select **Print all pages**. By default, the **Start page** and **End Page** fields will display the first and last page of your paper product. For more information, see the "Print Range" section in *Developing Paper Products*.



Accept the default value 0 in the **Start copy** and **End copy** fields. These defaults will print one copy of your paper product: the 0 copy. For testing, print only the 0 copy.

7. Click **Advanced Settings**.
8. In the Advanced Settings dialog, select the printing properties to use. For this example, choose the settings below.



9. After deploying your penlet, and deploying and printing your paper product, test your application. Tap on any of the printed controls to see that they work as expected. All the standard Livescribe controls should work like any other Livescribe paper products. You should be able to tap on the Record button to start Paper Replay. Tapping the Nav Plus should invoke the Main Menu.
10. Next, tap on the custom Active Area you defined for the Livescribe Dot Paper logo. It should play the Hello World wav file as you intended.

Penlet Configuration Editor

The Penlet Configuration Editor is the primary tool for configuring penlet resources and settings. You can access this tool from the Penlet Perspective either from **Livescribe > Penlet Configuration Editor** or by selecting the penlet project in the package Explorer and opening the context menu, and selecting **Penlet Configuration Editor**.

The editor has several tabs for configuring penlets: Properties, Image Resources, Audio Resources, Text Resources, and Advanced.

Properties Tab

Use this tab to configure basic penlet properties, including the name, classname, description, category (such as educational, entertainment), and version. Use this tab to set localized attributes too.

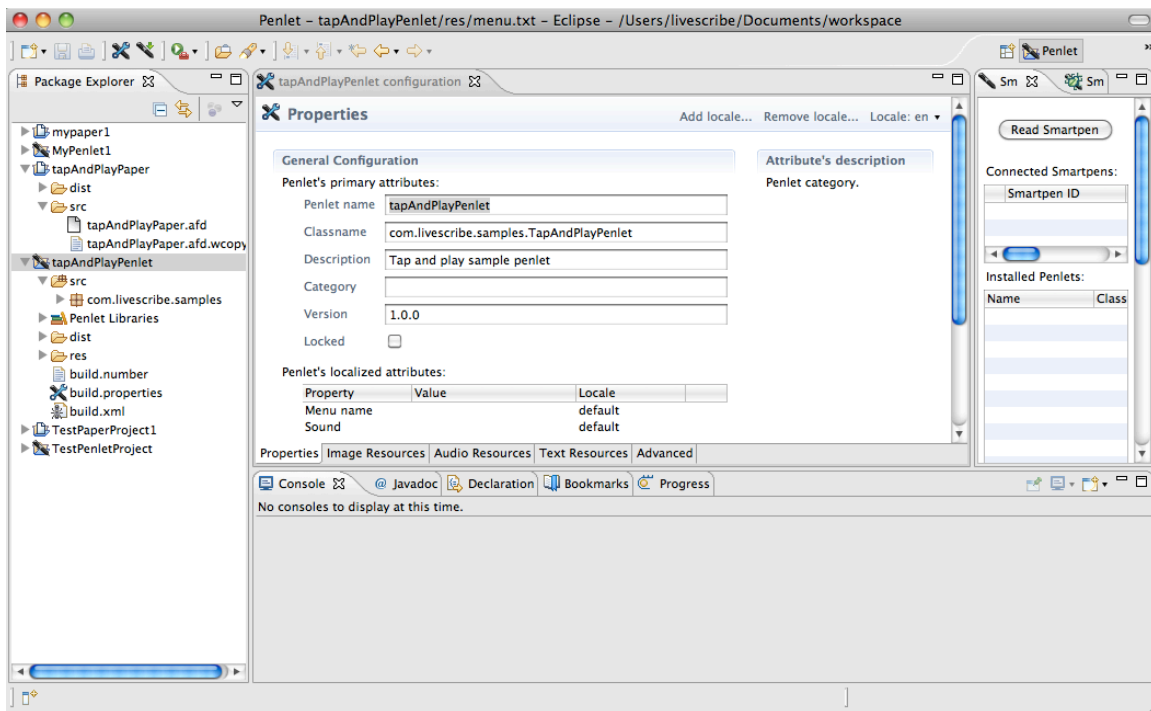
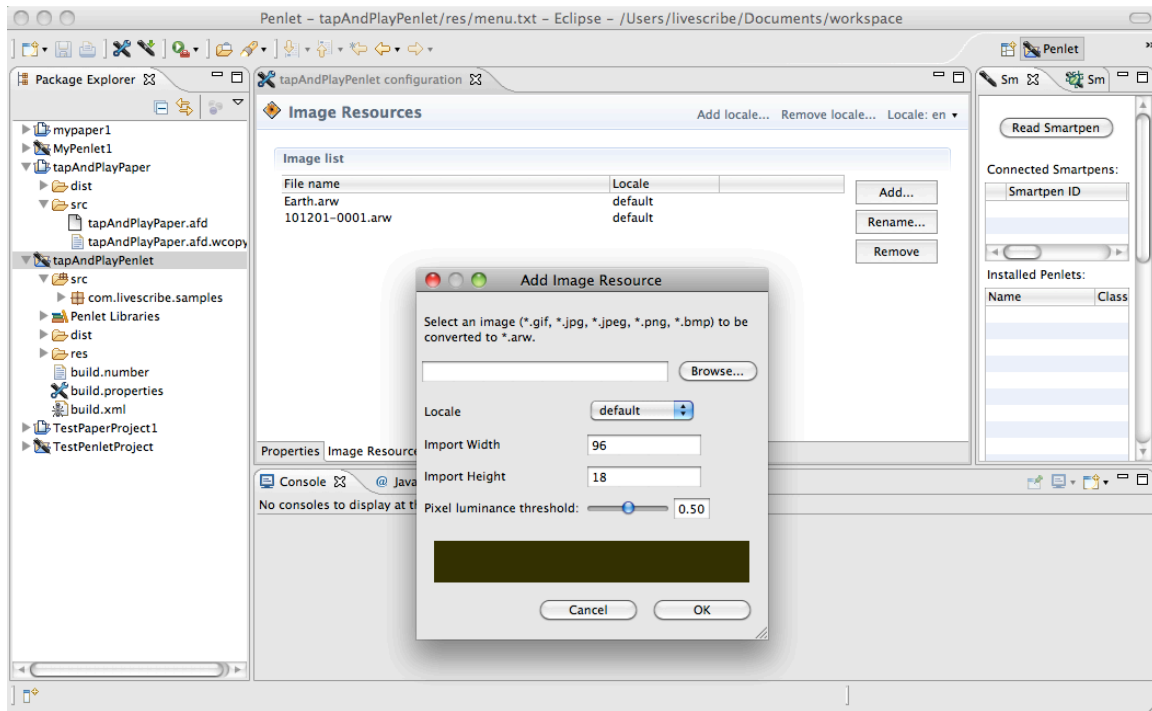


Image Resources Tab

Use this tab to create an image resource to show on the smartpen display. The image converter will transform an image you select into the required display format (ARW). See Preparing Smartpen Resources for more information.

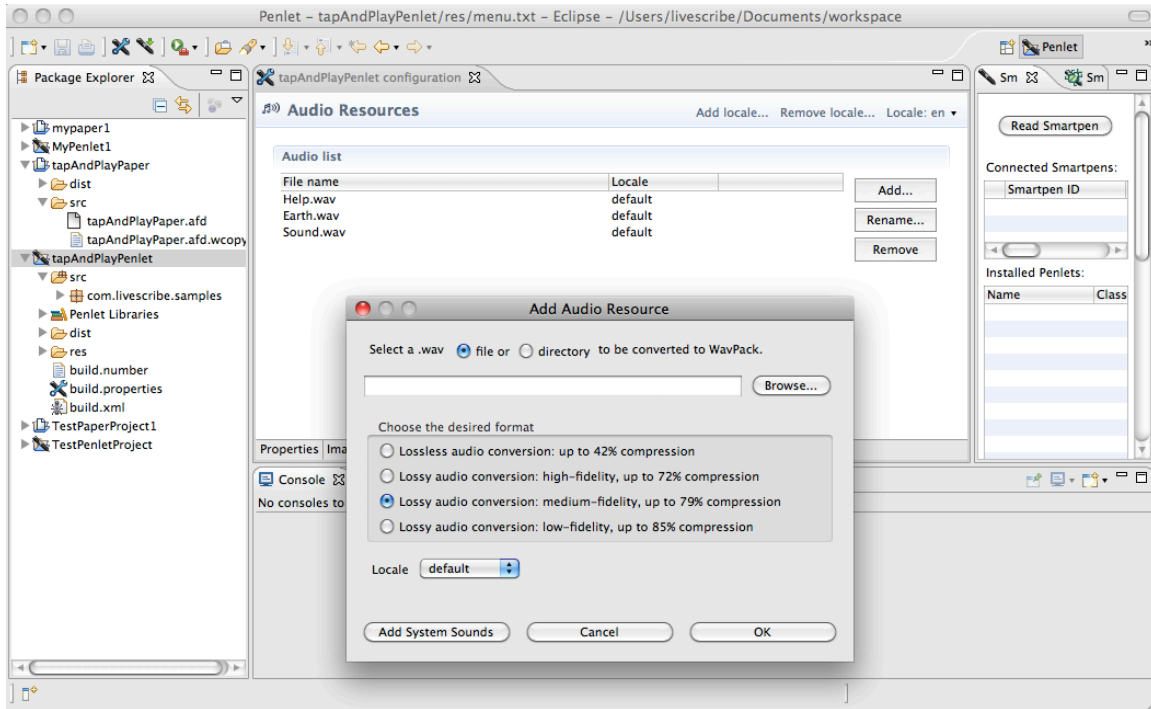
Getting Started with the Livescribe Platform SDK



Audio Resources Tab

Use this tab to create audio resources to play on the smartpen. The audio converter will transform an image you select into the required audio format. See [Preparing Smartpen Resources](#) for more information.

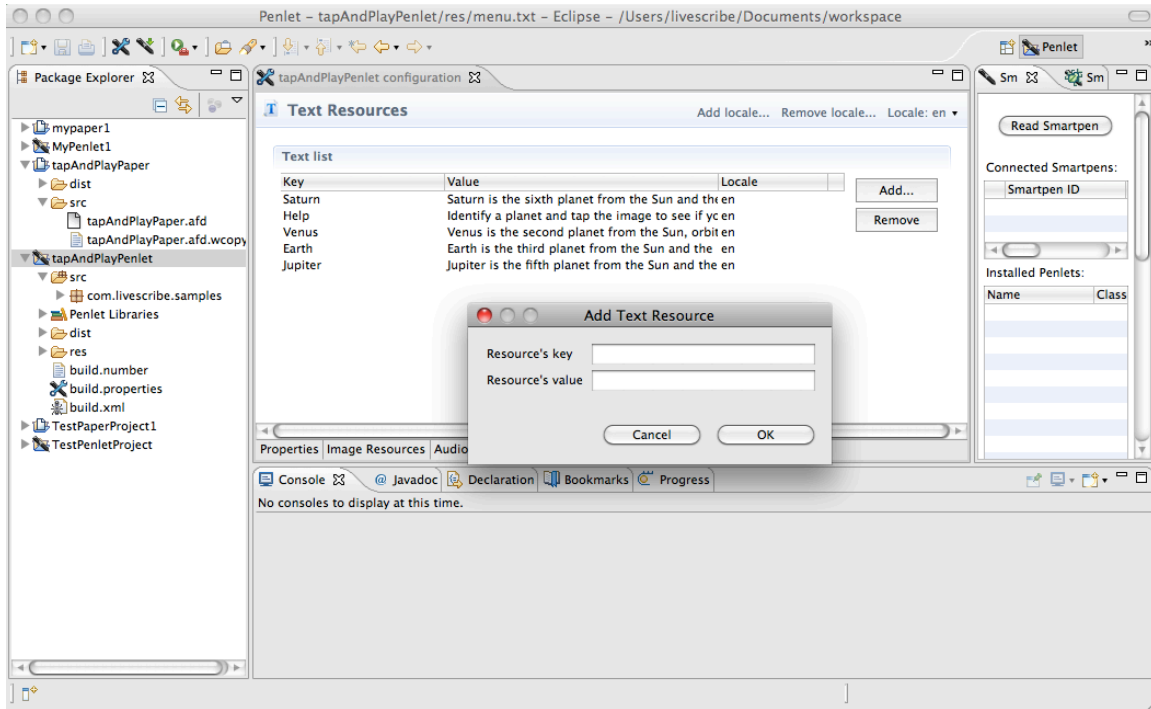
Getting Started with the Livescribe Platform SDK



Text Resources Tab

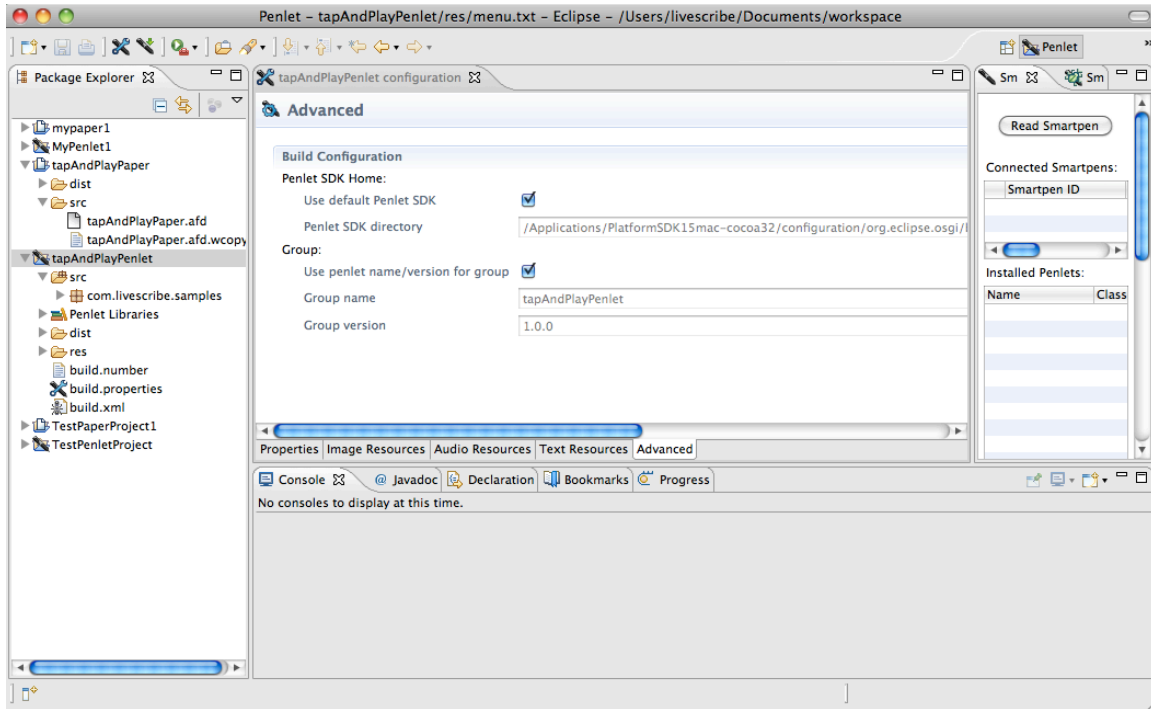
Use this tab to create text resources to show on the smartpen display. Enter text strings as key/value pairs. See [Preparing Smartpen Resources](#) for more information.

Getting Started with the Livescribe Platform SDK



Advanced Tab

Use this tab to define the penlet group name and version used when deploying applications to end users. See Building and Deploying Applications for more information. You can also configure the Penlet SDK Home folder to set up custom build environments.



Importing a Penlet Project

When you create a penlet project using Livescribe Penlet feature, the Eclipse workspace recognizes the project as being Livescribe-specific. However, projects created outside the Penlet feature, such as the Livescribe sample projects, won't be recognizable. This means you won't be able to run the deploy action on these projects or perform similar Livescribe-specific tasks.

For this reason, the Eclipse feature provides an import project wizard for converting externally-generated projects into Livescribe-specific projects.

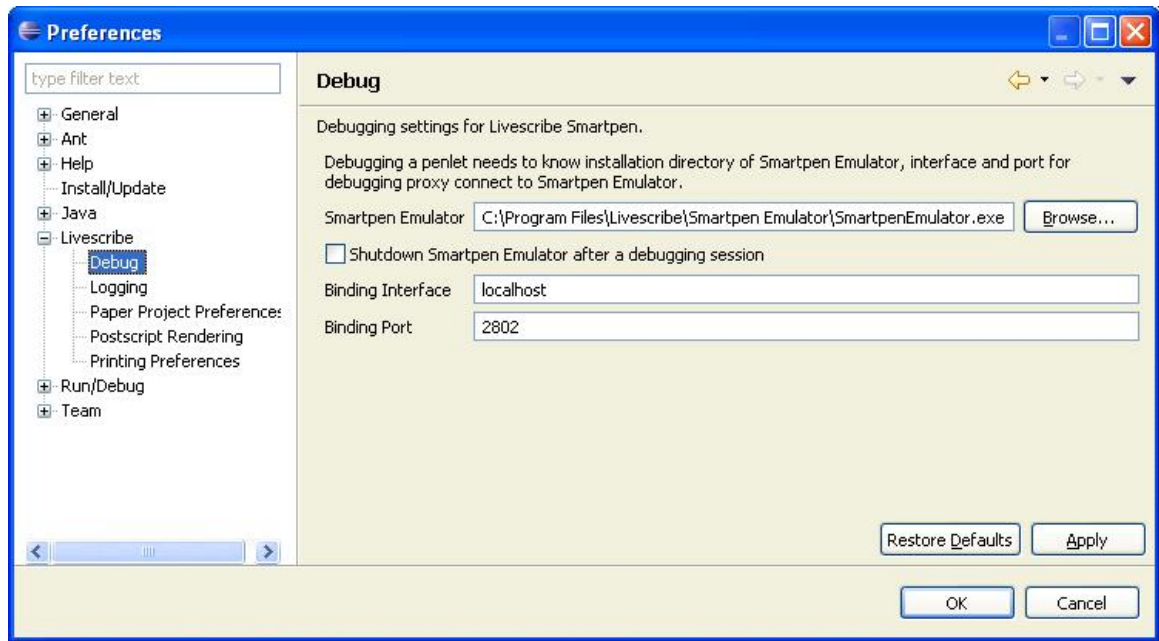
1. Select **File > Import**. . The Import wizard starts.
2. Select **Livescribe > Livescribe Penlet Project** from the Select page of the wizard.
3. Select the option labeled **Select archive file**. Browse to the root directory of your penlet project.
4. Click **Finish**.

Running and Debugging with Livescribe Smartpen Emulator (Windows Users)

If you are running Windows, you can run and debug your penlet and paper product using the Livescribe Smartpen Emulator.

Setting Up the Smartpen Emulator

1. Make sure you have already installed the Eclipse IDE and the Livescribe plugins. See the *Installing the Livescribe Platform SDK* guide.
2. If you have not done so already, download the Livescribe Smartpen Emulator from <http://www.livescribe.com/developer> and unzip it to a directory on your file system. (Downloading the emulator from the Livescribe Developer zone is very similar to downloading the Platform SDK.
3. Install the Livescribe Smartpen Emulator by running
`EmulatorInstallDir\Livescribe_Smartpen_Emulator_X.X\Installer\SmartpenEmulatorInstall.msi`,
where `EmulatorInstallDir` is the folder to which you unzipped the Livescribe Smartpen Emulator and `X.X` reflects the current version number of the emulator.
 - The installation is straightforward. If you desire more information, please consult *The Livescribe Smartpen Emulator User Guide* manual in the `Livescribe_Smartpen_Emulator_X.X\Docs` directory.
 - Accept the default installation directory. The installer will automatically enter the path to the emulator on the Debug page of the Preferences in Eclipse.
4. Verify that the installation has set the Debug preferences correctly.
 7. Select Window > Preferences > Livescribe > Debug.



8. On the Debug page, verify that the values are set as follows:

Property	Value
Smartpen Emulator	Path to the Livescribe Smartpen Emulator. Default value is: C:\Program Files\Livescribe\ Smartpen Emulator\SmartpenEmulator.exe
Binding Interface	Default value: localhost
Binding Port	Default value: 2802

Running a Penlet using the Smartpen Emulator

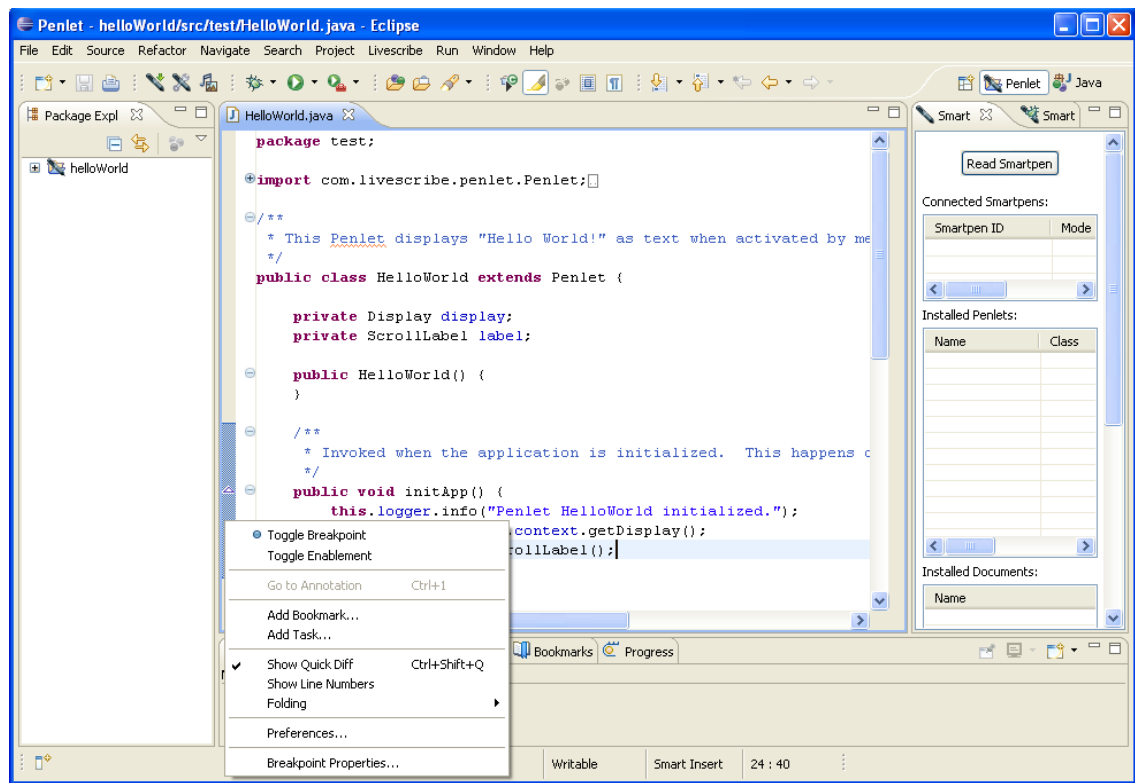
Windows users can deploy and run penlets directly from Eclipse using the Smartpen Emulator.


1. Create a penlet. (See [Creating an Open Paper Penlet](#) or [Creating a Fixed Print Application](#).)
2. Select a penlet project from the Package Explorer.
3. Right-click and choose **Run As > Penlet**. This deploys and launches the penlet to the Smartpen Emulator.
4. In the emulator, click Paper View, and click Open Paper Product.
5. In your workspace, locate the AFD in the dist folder for the penlet you are running.

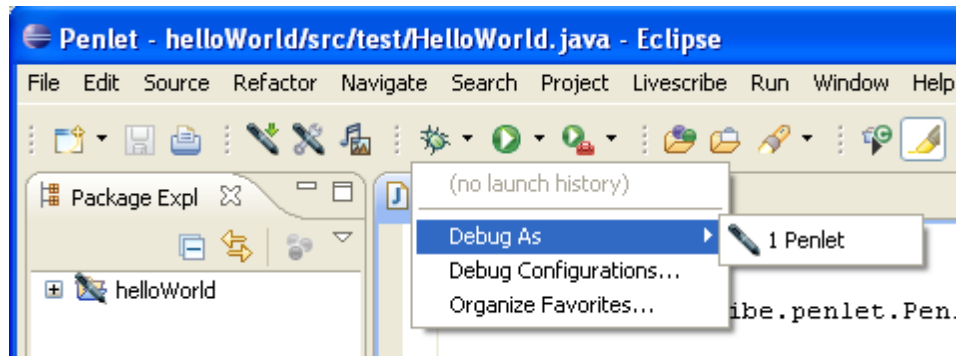
6. Click OK to open the paper product.
7. Test your application.

Debugging a Penlet using the Smartpen Emulator

1. Create a penlet. (See [Creating an Open Paper Penlet](#) or [Creating a Fixed Print Application](#).)
2. Set a breakpoint in the penlet. You can right click the gutter next to the code and select **Toggle Breakpoint**. (Alternatively, double-click the gutter to toggle a breakpoint.)



3. Click the down arrow on the debug toolbar icon  and select **Debug As > Penlet**.



4. The penlet automatically builds and the Livescribe Smartpen Emulator launches. The Debug panel reads: "Please wait while the system is starting up." When the emulator says "Updating...", the penlet is being deployed to the pen.
5. The various debug messages will display one line at a time. When they stop, you may continue.

6. In the Pen View window, click the **Paper View**  button. This opens the Paper View window.

- Drag the Paper View window into the center of the screen, if necessary.

7. In the Paper View window, click a button to open an AFD. The button you click depends on the kind of Livescribe smartpen application you have created:

- If you have an Open Paper application, click the **Open AFD Folder**



button, select one of the notebooks or flip pads, and click **OK**.

- If you have a Fixed Print application, click the **Open AFD File**



button, browse to your paper product's AFD, select it, and click **Open**.

8. Click on the NavPlus on the emulated dot paper to navigate through the menu system, locate your penlet, and run it.
9. When you run the penlet, a breakpoint should be hit. The text editor will display the line of code where the breakpoint was set.

Recommended Settings for Emulator Performance

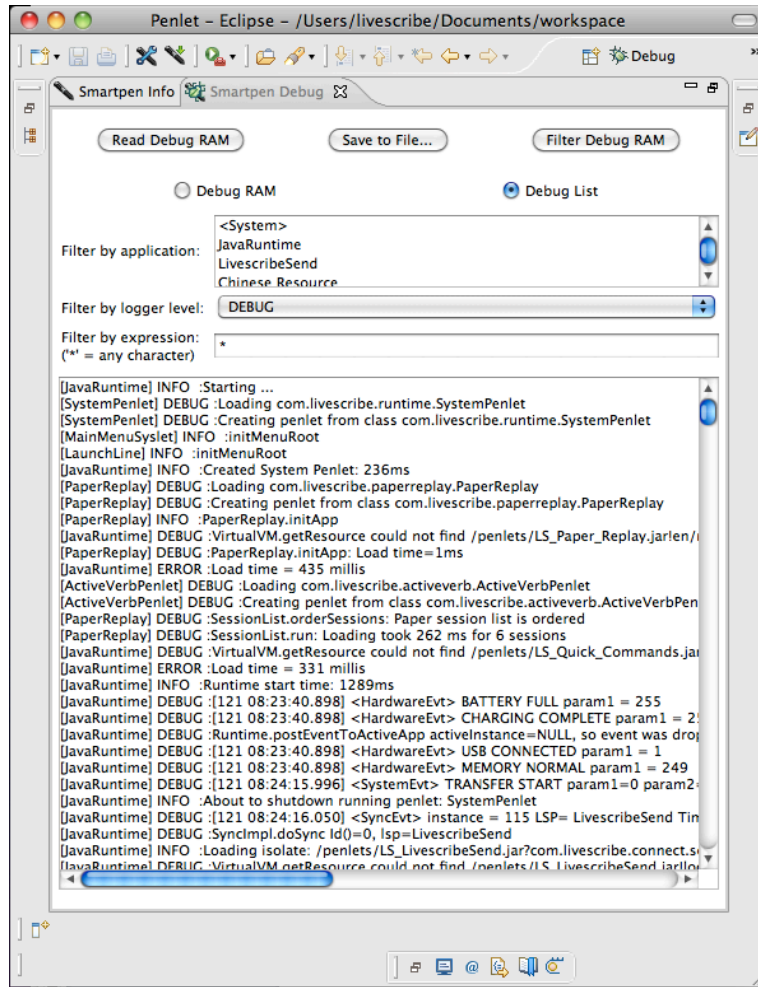
You may find that the Livescribe Smartpen Emulator is more responsive with the following settings:

1. Select **Window > Preferences > Java > Debug**.
2. On the Debug page, deselect the Suspend execution on uncaught exceptions setting.

Using the Smartpen Debug View

The Smartpen Debug View displays the debug messages stored on the currently-docked smartpen (or emulator). The information includes the most recent penlet- and system-related events. You can examine this output to discover the events that occurred during the smartpen's recent operation.

1. Select **Window -> Show View -> Smartpen Debug**. Alternatively, you may have to select **Window -> Show View -> Other -> Livescribe -> Smartpen Debug**. The Smartpen Debug view appears.



2. Click the **Read Debug RAM** button to display the smartpen's latest debug messages.
3. Optionally, click the **Filter Debug RAM** button to filter the currently displayed set of debug messages, without reading a new set.

You can filter the Debug RAM messages in three ways:

- *Filter by application*- If the smartpen's penlets are listed in the Smartpen Info view, you can filter messages that were output by a particular penlet.
 - **JavaRuntime** displays the log messages output by the smartpen's runtime.
 - **<System>** displays the log messages generated by the smartpen's low-level firmware.

Note: To select multiple applications to filter by, shift-click on each application name in the drop-down list. Names of the selected applications will display simultaneously in the field.

- *Filter by logger level* - The data log on the smartpen can have four levels of messages: **DEBUG**, **INFO**, **WARN**, and **ERROR**. This drop-down list allows you to limit displayed messages to one of these types. Each level is inclusive of the levels below it. For example, Debug includes Info, Warn, and Error messages. Error only includes Error messages.

Note: These log messages will be output to the data log only if the penlet code **calls** the appropriate methods (`debug`, `info`, `warn`, or `error`) of the `Logger` class.

- *Filter by expression* - If you enter text in the Filter by expression box and click **Read Debug RAM**, all debug messages containing that text will be displayed. The wildcard `*` matches 0 or more occurrences of any characters. Thus, `c*t` matches: `cat`, `cot`, `caught`, and `ct`.
4. Optionally, click the **Save to File** button to store the Debug RAM messages to a file on your computer. All data is saved as text.

Viewing the Debug List

In addition to viewing a smartpen's debug RAM, you can view the smartpen's Debug List. This list is a sequence of debug RAMs. Depending on the smartpen you are using, the list will include between four and 16 of the most recent debug RAMs.

Note: A new debug RAM is added to the list when the smartpen is started. The debug list will not contain the debug RAM that is currently being written by the smartpen. You can only read this using the Debug RAM option.

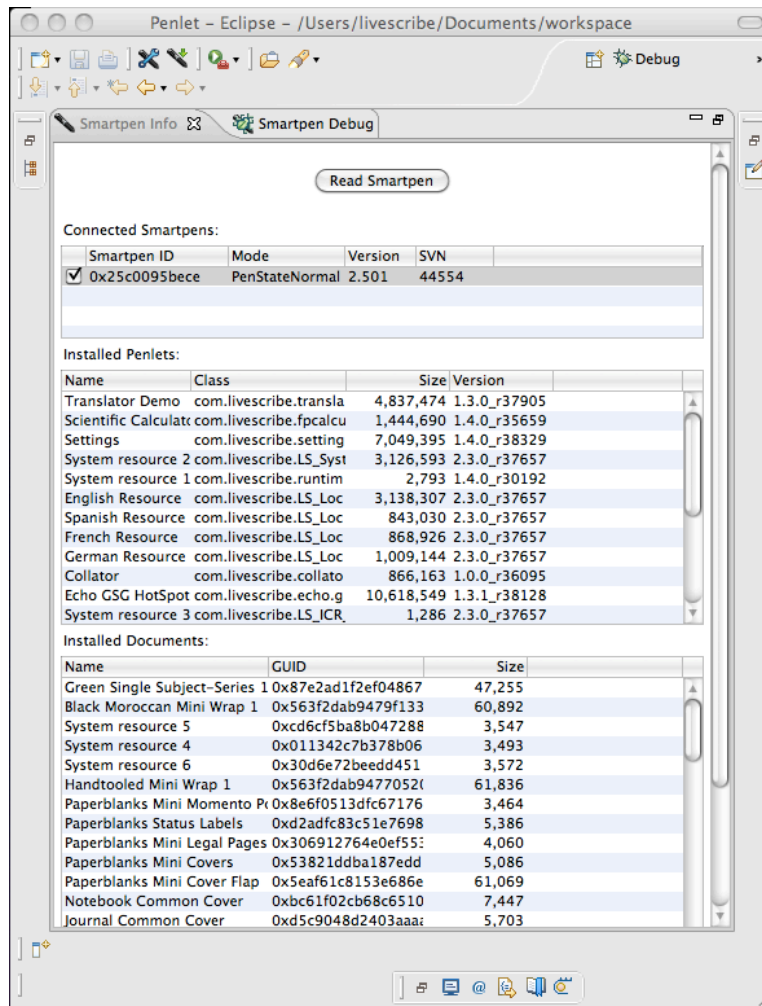
1. In the Smartpen Debug panel, select the **Debug List** radio button.
2. Click the **Read Debug RAM** button to get the Debug List.

As with Debug RAM, You can filter this list or save it to a file.

Smartpen Info View

You can view information about the currently-docked smartpen, including its installed penlets and paper products (AFDs).

1. Select Window -> Show View -> Smartpen Info. (Alternatively, you may have to select Window -> Show View -> Other -> Livescribe -> Smartpen Info.) The Smartpen Info view appears.



2. Click the **Read Smartpen** button in the Smartpen Info view. If no smartpens are connected, the first table will respond with the message No Smartpens Connected. If a smartpen is connected, its penlet classes and penlet documents (AFDs) will be listed in the Installed Penlets and Installed Documents tables.

You can also use these tables to uninstall your penlets and documents (AFDs) from the smartpen.

- To uninstall one of your penlets, right-click its name in the Installed Penlets table and select **Remove penlet "MyPenletName"**. For instance, if you right-click on the HelloWorld penlet, the context menu item says: Remove penlet HelloWorld.

Note: You cannot uninstall the system penlets or the penlets that come bundled with the Livescribe smartpen.

- To uninstall any document, right-click its name in the Installed Documents table and select **Remove Document Name**. For instance, if you right-click on the Tutorial document, the context menu item says: Uninstall Tutorial.

Note: Be careful when removing documents. If you uninstall one of the Livescribe notebooks or notepads, such as Lined Journal 3, your smartpen will no longer recognize that Livescribe paper product.

Setting Up Library Projects in Eclipse

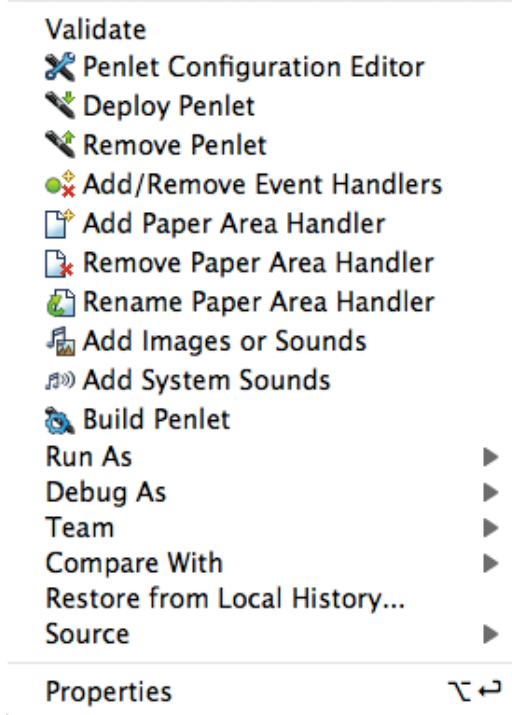
You can set up library projects to organize your code using Eclipse projects.

1. Go to Java Build Path/Project tab
2. Add referenced projects. The Platform SDK uses the source paths from these referenced projects adds them to the java.src.path variable in build.properties file.

Using the Livescribe Context Menu

The Livescribe context menu provides commands for penlet development and deployment. To access this menu, right-click the **top node** of a penlet project in one of the resource navigation views: Project Explorer, Package Explorer, and Navigator.

A long context menu appears. Near the end are the menu items pertaining to development for the Livescribe Platform.



The Livescribe items in the context menu are:

Menu Item	Description
Penlet Configuration Editor	Opens editor for modifying penlet properties and adding/removing audio, image, and text resources. See A Basic Tap-and-Play Application for examples of using the Penlet Configuration Editor.
Deploy Penlet	Uploads the penlet JAR to the smartpen.
Remove Penlet	Uninstalls the penlet from the smartpen.
Add/Remove Event Handlers	
Add Paper Area Handler	
Remove Paper Area Handler	
Rename Paper Area Handler	
Add Images or Sounds	

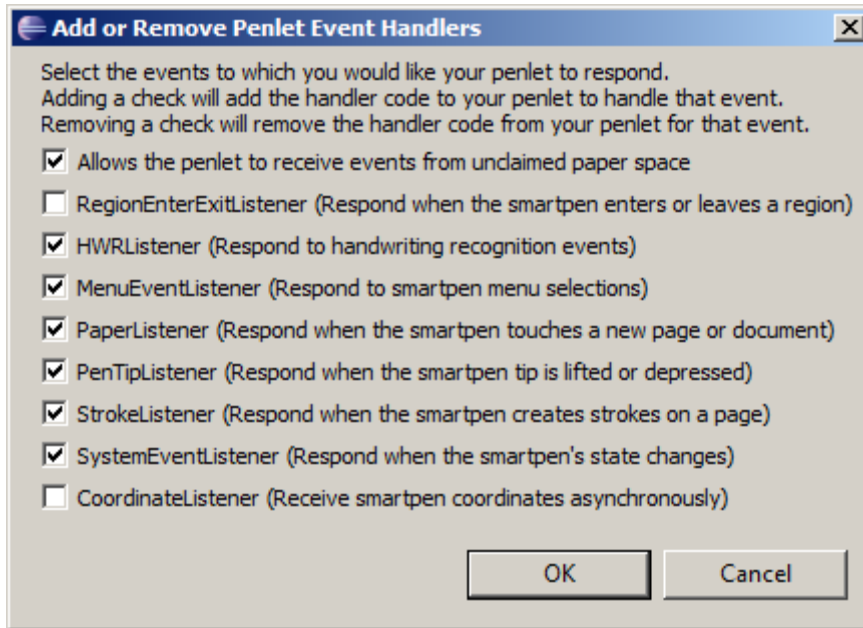
Add System Sounds	
Build Penlet	Creates a deployable JAR file for the project.
Run As > Penlet	Runs the currently-selected penlet. If you are running Windows and have installed the Smartpen Emulator, this command launches the emulator and runs the penlet on it.
Debug As > Penlet	Runs the currently-selected penlet in Debug mode. If you are running Windows and have installed the Smartpen Emulator, this command launches the emulator and runs the penlet on it.

Add/Remove Event Handlers

This dialog allows you to add event listeners to and remove event listeners from the source code of your penlet project. In Java, an event listener is an interface that contains one or more methods known as event handlers.

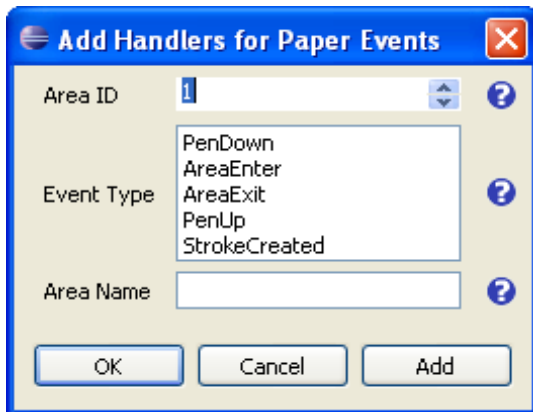
The checkbox options in the Add/Remove Event Handlers dialog are identical to the Event Configuration page of the Livescribe Penlet Project Creation Wizard. See [Event Configuration Page](#).

To add an event listener, check the box in front of it. To remove an event listener, check the box in front of it. Then click OK. All event listeners you checked will be added to your source code. All event listeners you left unchecked will be removed from your source code. (If an event listener has already been added, checking its box will have no effect on your source code.)



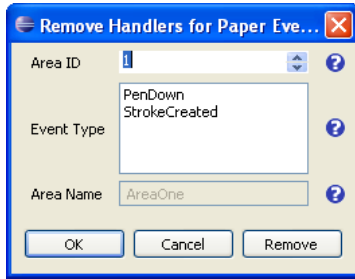
Add Paper Area Handler

This dialog allows you to add one or more event handlers for an area associated with FP (Fixed Print) paper. The code generator inserts appropriate code in your source to support an area. Each Area ID and Area Name must be unique within the penlet. The Area ID is a positive integer value. The Area Name is a string conforming to Java naming conventions (e.g., no spaces allowed).



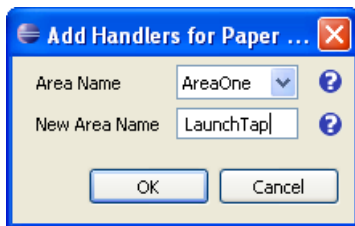
Remove Paper Area Handler

This dialog allows you to remove one or more event handlers for an area associated with FP (Fixed Print) paper. The code generator removes code from your source that it originally inserted to support the event handler(s).



Rename Paper Area Handler

This dialog allows you to rename an event handler for an area associated with FP (Fixed Print) paper.



Add Images or Sounds

The Add Images or Sounds command allows you to add one of the following to your penlet project:

- Image for display on the smartpen OLED
- Sound to be played on your penlet project.

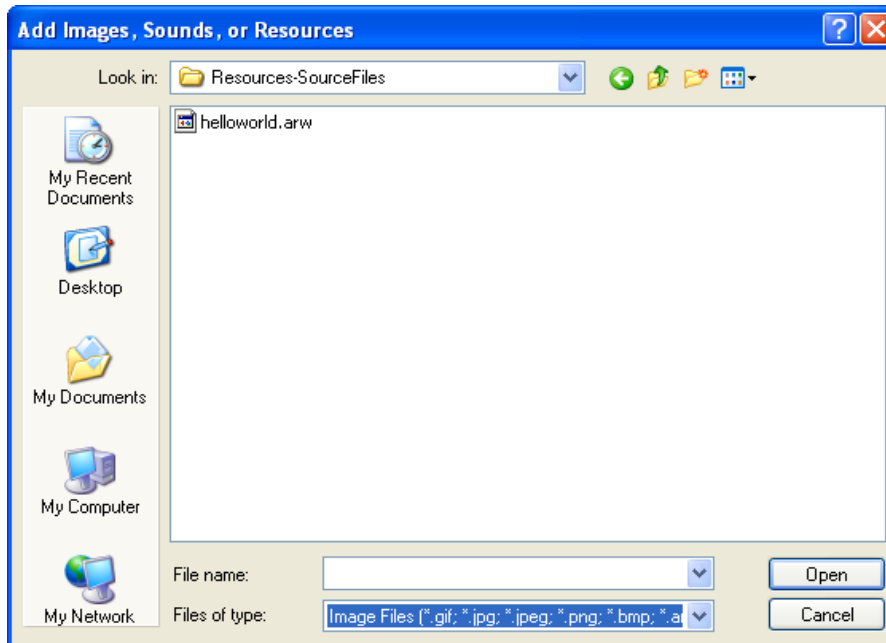
To access this command:

1. Right-click the top node of the project in Package Explorer.
2. Select **Add Images or Sounds** from the context menu.

3. Select **Image Files or Audio Files** from the Files of type drop-down box.
4. Navigate to the file(s), highlight, and click **Open**.

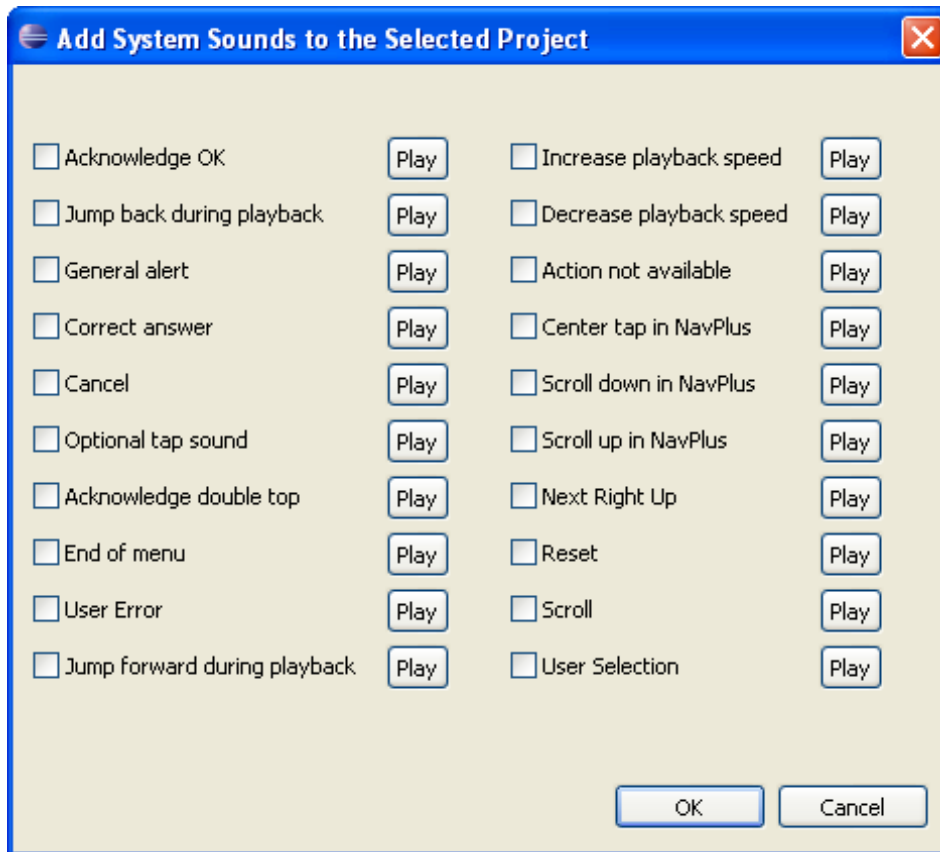
Images will be added to the `res/images` directory and sounds will be added to the `res/audio` directory of the penlet project.

For additional information about adding resources to your penlet, see [Preparing Smartpen Resources](#).



Add System Sounds

The fourth option, **Add System Sounds**, lets you insert additional audio files (*.wav) into the currently selected project. The following figure shows what the **Add System Sounds** dialog box looks like.



Click the **Play** button to hear the sound produced by the audio file. To choose one or more system sounds, click the corresponding checkboxes in the dialog. Then click **OK** to add them to the currently selected project. The selected WAV files will be placed in the `/res/audio` directory.

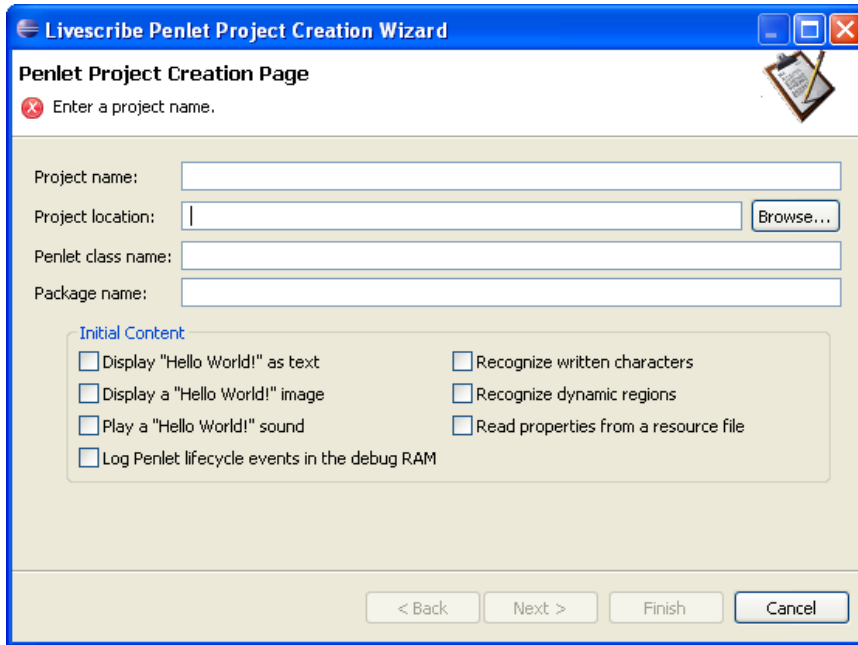
Using the Livescribe Penlet Project Creation Wizard

Launch the Livescribe Penlet Project Creation Wizard by selecting **File > New > Livescribe Penlet Project**. The wizard consists of three pages:

- [Penlet Project Creation Page](#)
- [Event Configuration Page](#)
- [Application Configuration Page](#)

Penlet Project Creation Page

This is the first page of the Livescribe Penlet Project Creation Wizard. It consists of two halves.



In the top half, you specify the project name, project location on your file system, and the package name.

In the bottom half, you can specify what kind of functionality you would like the automatic code generator to insert in your penlet. The wizard will generate the corresponding listeners and stubs for the associated event handlers for you.

This generated code provides simple actions. You can learn some basics of penlet programming by studying the code. Later, you might want to start with the default code of these event handlers and modify them to fit the needs of your new penlet.

The following table summarizes the options of the Initial Content section (bottom half) of the Penlet Project Creation Page:

Option	Inserts code for:
Display "Hello World" as text	Displaying "Hello World" on the OLED of the

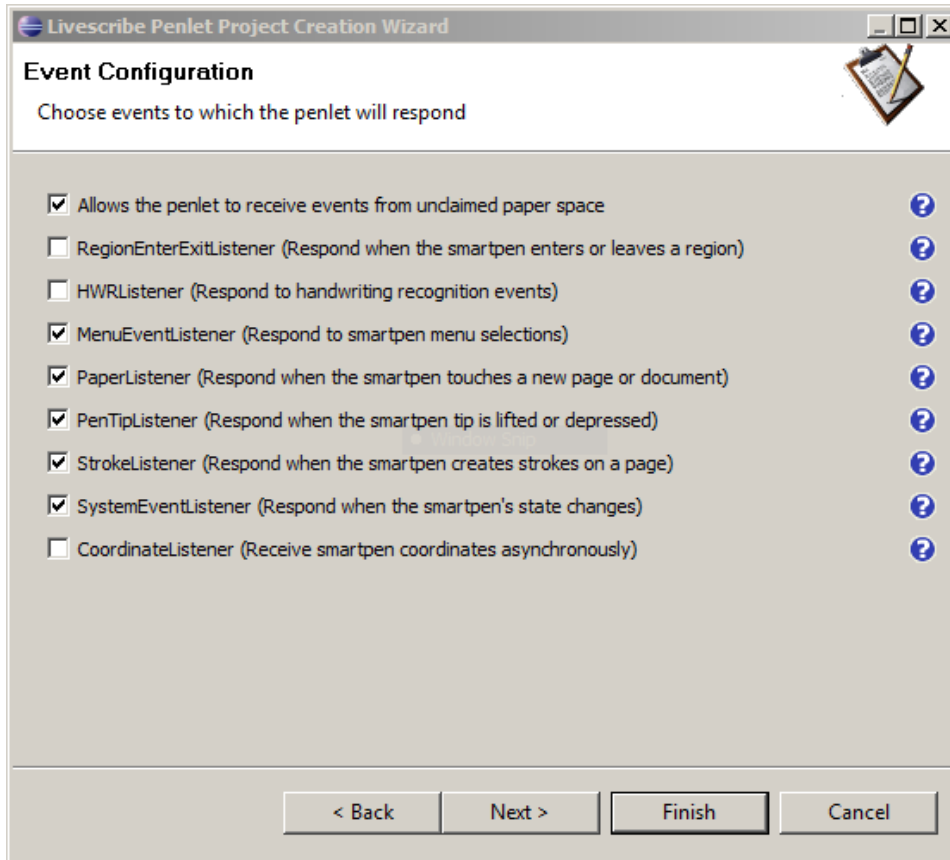
Getting Started with the Livescribe Platform SDK

	smartpen. Text displays when penlet is launched.
Display a "Hello World" image	Displaying an image (in ARW format) on the OLED of the smartpen. Image displays when penlet is launched.
Play a "Hello World" sound	Playing a sound on the smartpen when the penlet is launched.
Log Penlet lifecycle events in the debug RAM	<p>Outputting debug statements that identify the execution of life cycle events in the penlet. Output is to the Smartpen debug RAM. The debug RAM may be viewed via the Debug View in Eclipse.</p> <ol style="list-style-type: none">1. Deploy penlet.2. Use a Nav Plus to navigate through the smartpen's main menu.3. Launch the penlet.4. In the Smartpen Debug View, select LifeCycleEvents from the Filter by application drop-down box.5. Click Read Debug RAM.
Recognize written characters	Recognizing characters written by the user on Open Paper.
Recognize dynamic regions	Creating a region when the user draws a shape on Open Paper. Subsequently, when the user interacts with the shape, the penlet is activated.
Read Properties from a resource file.	Writing a single name=value pair (CONFIGDATA=Hello World!) to the config.txt file. When launched, the penlet reads this property and displays the name and its value on the smartpen OLED.

Event Configuration Page

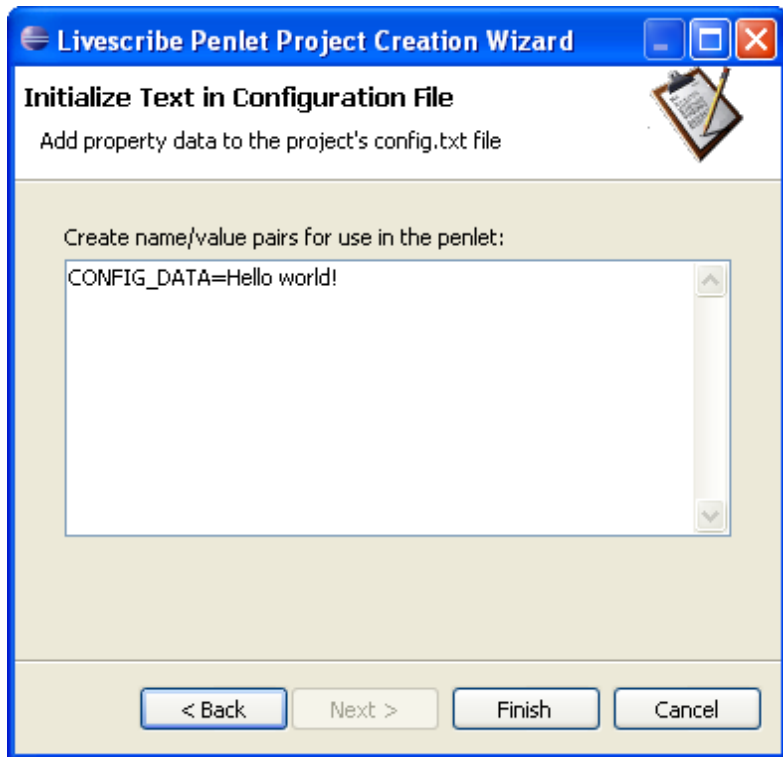
The second page of the **Livescribe Penlet Project Creation Wizard** is the Event Configuration page. Checking an item will create event listeners and event handler stubs that you can implement later.

Option	Listener Adds These Methods:
Can receive events from unclaimed paper space	canProcessOpenPaperEvents and sets return value to true.
RegionEnterExitListener	regionEnter regionExit
HWRLListener	hwrCrossingOut hwrError hwrResult hwrUserPause
MenuEventListener	handleMenuEvent
PaperListener	onNewDocument onNewPage
PenTipListener	penDown penUp singleTap doubleTap
StrokeListener	strokeCreated
SystemEventListener	handleSystemEvent
CoordinateListener	coordinate coordinate failed hover hoverAway whitePaper



Initialize Text in Configuration File

This page of the wizard appears only if you chose **Read properties from a resource file** on the Penlet Project Creation Page. Specify name-value pairs, which will be written to the `config.txt` file. The penlet reads this file when it is launched. And your code can read these pairs at runtime.



Intelligent Character Recognition (ICR) Configuration Page

The Intelligent Character Recognition (ICR) Configuration page of the Livescribe Penlet Creation Wizard appears only under one of the following circumstances:

- You select **Recognize written characters** in the [Penlet Project Creation Page](#).
- You select **HWRLListener (Responds to handwriting recognition events)** in the [Event Configuration Page](#) of the Livescribe Penlet Project Creation Wizard.
- You select **HWRLListener (Responds to handwriting recognition events)** in the [Add/Remove Event Handlers](#) dialog.

The Intelligent Character Recognition (ICR) Configuration page allows you to specify values that will limit the scope of writing the ICR engines will recognize. You choose to specify a subset of the total possible characters to improve:

- Accuracy. For example, an "I" can look like a numeral "1." If you limit your penlet's recognition capabilities to only numerals, you have increased the chances for accurate recognition.
- Performance. You can optimize the ICR engine's speed by limiting the set of characters if needs to recognize.

You specify a custom lexicon to restrict the ICR engine to recognize only these terms—which can improve both accuracy and performance.

Option	Action
The minimum delay (ms) interpreted as a pause	Specifies the period of time without the user making new strokes that the ICR engine will interpret as a pause (and usually, the end of the stroke). In milliseconds.
Configure the penlet's character	(See table below)

Getting Started with the Livescribe Platform SDK

subset	
Use Custom lexicon	Words you specify here will be inserted in a custom lexicon. No other words will be recognized.

The following table describes the available subsets of the entire character set. You can select only one of these.

Option	You specify a subset of the total possible characters
Alphabetic characters and numbers	Alphabet characters and numerals only
Alphabetic characters only	Alphabet characters only (both upper and lower case)
Numbers only	Numerals (1 2 3, etc.) only—no decimal points, commas, nor positive/negative signs allowed.
Custom subset (Add each character)	Only the characters you enter in the field
Lower case	Alphabet characters—lower case only.
No subset (Use entire alphabet)	Uses all alphabet characters, numerals, punctuation

Application Configuration Page

This is the third page of the Livescribe Penlet Project Creation Wizard. It allows you to add your penlet to the smartpen's main menu, control penlet ID and version number, and configure the penlet's behavior in the smartpen's main menu. When deployed, your penlet becomes part of the MyApps submenu of the smartpen's main menu.

Note: This is not the same as creating a single- or multi-dimensional menu for your penlet.

Getting Started with the Livescribe Platform SDK

The screenshot shows a Windows-style dialog box titled "Livescribe Penlet Project Creation Wizard". The subtitle is "Additional Configuration" with a subtext "Provide information about the penlet application". The main area contains several input fields: "Application identifier:" with the value "HelloWorld", "Application version number:" with the value "1.0.0", and a checked checkbox "Add the application to the Pulse smartpen menu". Below this is a "Menu Configuration" section with "Menu name for application:" set to "HelloWorld", a "Sound to play when menu item appears:" field with a "Browse..." button, and an unchecked checkbox "Select system sound". At the bottom are four buttons: "< Back", "Next >", "Finish", and "Cancel".

The Application Configuration page appears. On this page you can modify:

Field Label	Explanation	Default Value
Application identifier	Name of the built penlet	The built penlet has the same name as the project
Application version number	The version number of the penlet	Penlet version is 1.0.0.
Add the application to the smartpen menu	Whether the penlet appears on the smartpen's main menu.	Penlet will appear as an item on the smartpen's main menu
Menu name for application	Name that appears in the smartpen's main menu for the penlet.	Penlet's name on the main menu is the same as the project name
Sound to play when menu item appears	Custom audio file that plays when the penlet's menu item rolls into view in the smartpen's main menu.	No audio file.

Select system sound	Play the default system sound when the penlet's menu item rolls into view in the smartpen's main menu.	Default value: unchecked. Note: The audio we are concerned with here is played only when the user is scrolling through the smartpen's main menu. When the penlet's name rolls into view in the OLED, the sound plays. Not all penlets have this menu sound.
---------------------	--	---

Preparing Smartpen Resources

The Livescribe smartpen can play audio files and display images, but only if the resource files are suitably formatted. Image files must be converted into the Livescribe-specific ARW format, and audio files must be either WAV (*.wav) files or WavPack (*.wv) files.

The Livescribe IDE provides tools for converting resources. You can access these tools using the Penlet Configuration Editor. You can launch this editor using the context menu or by choosing **Livescribe > Configuration Editor** menu item.

Adding Smartpen Display Images

Use the image converter to format images (*.gif, *.jpg, *.png, and *.bmp) to the Livescribe ARW format.

1. From the configuration editor, go to the **Image Resources** tab.
2. Click **Add**.
3. Click **Browse** to bring up a file selection dialog
4. Click **OK** to save the converted image into the currently-selected project.

Below the file selection text box, the preview canvas shows what the converted image will look like. If the original image is a color image, each color in the image will be converted to a luminance value between 0.0 and 1.0.

By default, pixels with a luminance greater than 0.5 will be colored white, while those with a luminance less than 0.5 will be colored black in the ARW image. You can adjust the luminance threshold by adjusting the scale.

Editing Smartpen Display Images using the ARW Editor

ARW is a proprietary monochrome format used specifically with the Livescribe smartpen. You can use the Image Converter in the Penlet Configuration Editor to convert existing images to ARW, but you cannot use the converter to edit the images. Instead, you can use the ARW Editor.

Note: The ARW format supports images up to 256x256 pixels. The Pulse and Echo smartpen screens are limited to 18x96 pixels, but will scroll horizontally through the entire 256 pixel image if it is larger than 96. Only the first 18 pixels are used in the vertical dimension.

To run the ARW editor:

1. Add an ARW using the Penlet Configuration Editor.
2. Locate the ARW image in the `res/images` directory of your penlet project.
3. Double-click on an ARW file in the Package Explorer. The ARW editor appears in a separate window in the IDE.



4. The ARW editor is a 1-bit pixel editor. By clicking in the editor, you can convert dark pixels to light or vice-versa. You can also drag the mouse over the image to color additional pixels in the area. In addition to drawing pixel-by-pixel, you can draw lines, ellipses, or rectangles. Use the right-click context menu to see these other editing options.
5. When you close the editor window, choose **Save** to preserve your edits.

Converting Audio

Use the audio converter to change your audio files into one of the supported audio formats: WAV (*.wav) and WavPack (*.wv).

The WavPack format provides the same quality as WAV, but the converted files are approximately half the size when converted losslessly. With certain lossy compression modes, the resulting compression can be up to 85%.

To perform audio conversion:

1. From the configuration editor, go to the Audio tab.
2. Click the Browse... button and navigate to the WAV file you intend to convert.
3. Select one of the conversion choices.
4. Click **Save File** to add the converted file to your penlet project.

Note: Not all audio files can be converted or played on the Livescribe smartpen. To be convertible, all WAV files must be sampled at 16kHz, with 16 bits per sample.

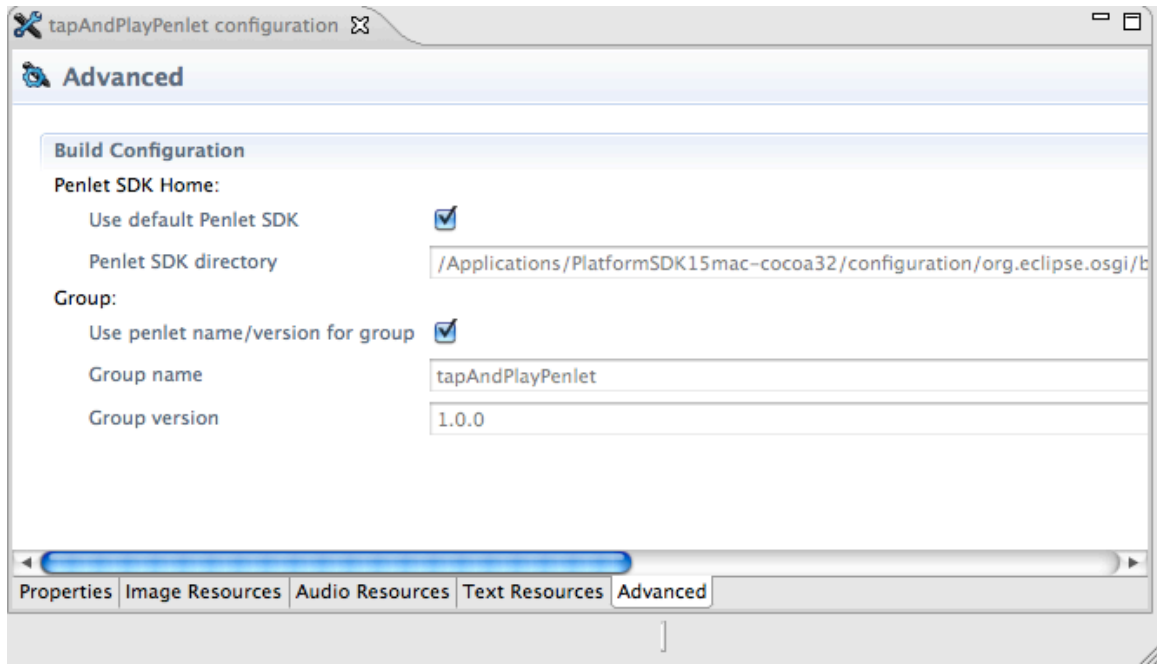
Building and Deploying Applications

As a developer, you deploy Penlets to smartpens as Java archive (JAR) files. Similarly, you deploy Paper Products to smartpens as AFD files.

To build and deploy a penlet:

1. From the Package Explorer, select a penlet project.
2. Right-click to view the context menu and choose **Penlet Configuration Editor**.
3. Go to the **Advanced** tab and enter build configuration information. You can specify an alternative Penlet SDK Home folder, and a group name and group

version for your application.



4. From the context menu, choose **Build Penlet**. The SDK will build penlet JAR and a corresponding bundle (.bnd) file.
5. To deploy the JAR to your smartpen or emulator, from the context menu, choose **Deploy Penlet**.

Note: Livescribe deploys applications to end users as digitally-signed bundles (.bnd files). Although you can build bundles using the SDK, you cannot deploy them without prior approval from Livescribe.