

RAGE

Tutorials

Resource Plugins (File Loaders)

Software & Documentation © The RAGE Team 2005

Version Relevance: JGE Release 0.1+

CONTENTS

Are you reading the right tutorial?	3
Basics	4
Example Loader	5
Other Utilities	6

ARE YOU READING THE RIGHT TUTORIAL?

This tutorial is intended for writing a resource loader that will be used to load resources into the ResourceManager of JGE. It does NOT demonstrate how to use the resource system in an application.

BASICS

A ResourceLoader is given an InputStream which it must take its data from to construct an object which is the resource.

A ResourceLoader need only implement a default constructor and a single function:
`+loadResource(java.io.InputStream):java.lang.Object`

The return of the function is the resource loaded from the InputStream. Resources can be any object of type java.lang.Object. Typically there are generic objects that loaders should use for particular types of files.

<i>File Type</i>	<i>Resource Object Type</i>
Image/Photo (JPG/GIF/etc)	jge.hal.render.core.Image
Audio	jge.hal.sound.Sound
Font	jge.widget.font.Font
Text	java.lang.String

The ResourceLoader should NOT close the InputStream when it is finished with it.

EXAMPLE LOADER

The code in this example demonstrates loading a Sound object from a .wav file.

The loader needs to be derived from `jge.kernel.resource.ResourceLoader`. The constructor passes up to the superclass the extension this loader handles. i.e. This loader can load .wav files (e.g. `mysound.wav`).

```
public class WAVSoundLoader extends ResourceLoader
{
    public WAVSoundLoader()
    {
        super("wav");
    }
}
```

We then need to add the `loadResource` function which does all the loading of the resource. The `ResourceInputStream` utility is being used to simplify the process (see next section). **(Note: This is a simplified example, and would not actually load a .wav file, you would need to read much more from the file to make a stable loader.)**

```
public Object loadResource(InputStream datastream)
{
    Sound result = null;
    ResourceInputStream ois = null;

    try
    {
        ois = new ResourceInputStream(datastream);
    }
    catch(Exception e) { e.printStackTrace(); }

    //get data about the file
    int samples = ois.readInt();
    int chunkSize = ois.readInt();

    //load data - loop to make sure it read everything
    byte data[] = new byte[chunkSize];
    int readSize = 0;
    while((readSize < chunkSize))
    {
        readSize += ois.read(
            data, readSize, chunkSize-readSize);
    }

    result = new Sound(
        samples, SoundFormat.STEREO_16BIT, data);

    return result;
}
```

OTHER UTILITIES

A class `jge.kernel.resource.ResourceInputStream` provides extra functionality for reading `InputStreams`. It wraps the `InputStream` and gives access to pieces of its data in different formats.

You can request from the stream the next int, and it will compile together the byte into an int and return its value. Similar functions exist for other file formats including `Strings`.

It also allows you to read chunks of data using an offset and reading a particular length into an array.

