

RAGE

Tutorials

Properties Management System

Software & Documentation © The RAGE Team 2005

Version Relevance: JGE Release 0.1+

CONTENTS

Are you reading the right tutorial?	3
What are properties?	4
When to use properties	
Rage system properties	
API	5
Required Imports	
Classes	
Interfaces	
Example	

ARE YOU READING THE RIGHT TUTORIAL?

This tutorial will attempt to explain the properties management system in the RAGE Kernel. It also explains the need to set RAGE system properties in an application. This document is accompanied by a sample program, PropertiesSample.

WHAT ARE PROPERTIES?

Properties in RAGE are essentially managed global variables. The properties can be used by applications written using RAGE, and are also used by RAGE sub-systems.

When to use Properties

Properties are most useful in situations where certain variables are set externally (i.e, from a file). The PropertiesManager of RAGE provides safe mechanisms to load Properties from a file.

Loading variables externally allows for a much more flexible and modular program. Properties that are loaded externally allow for radical differences in the programs behaviour, without the need for recompilation. There are literally hundreds of possibilities. For example, the program might load a set of properties that:

- Give filenames for which textures to load.
- Map keys to game actions.
- Contain seeds for random Perlin noise functions.
- Determine the initial state of game objects.
- Determine the behaviour of game objects (e.g, velocity, mass, response to collision, “state of mind” for artificially intelligent agents...).

Properties are also useful when multiple classes need to directly observe changes in the same variable. The Properties system of RAGE provides a simple yet effective observer pattern, allowing any class to be immediately notified of changes to any property in the registry.

RAGE System Properties

You may have already noticed that virtually all RAGE programs will make some reference to the PropertiesManager at some point. This is because a lot of the RAGE sub-systems make use of the PropertiesManager in order to achieve an extra level of platform independence. For example, the Renderer looks for a property in the PropertiesManager that tells it which implementation class to use.

When using RAGE, it is your responsibility as a programmer to set the appropriate properties for the RAGE sub-systems. If the systems can not find the required properties, they most likely will not run, and throw some sort of exception, crashing your program. The main areas to look out for are the Kernel and all the HAL subsystems (i.e, Input, Renderer, Sound, etc). Have a look at the JavaDocs for these systems to see exactly which properties they require.

API

The source code for the Properties Management System is fully documented - for a complete specification of the API for the system, please refer to the JavaDocs.

Required Imports

The Properties Management System of RAGE lies within the Kernel, inside a single package. Thus, the entire API can be imported with a single statement:

```
import jge.kernel.properties.*;
```

Classes

There are two public classes in the package:

- `PropertiesManager` – essentially encapsulates a `java.util.Properties` object, with the main difference that it is a `Singleton`.
- `PropertiesObserverManager` – a container for all active `PropertiesObservers`.

Interfaces

- `PropertiesObserver` – implemented by any class that wishes to be notified of changes in the `PropertiesManager`.

Example

See the attached sample program, `PropertiesSample`, to see how to use the Properties Management System in your programs.

