Jukebox: Configuration

Table of contents

Copyright © 1990 Vadim Tkachenko All rights reserved.

1. Introduction

The configuration has to comply to two basic requirements:

- Richness of the configuration object
- Simplicity and speed of access to the configuration

Historically, the configuration component was started as a simple UNIX-style configuration file reflection. Later, when XML arrived, and the fact that XML files represent a perfect configuration media, the XML configuration reflection was added. Today, the XML configuration is preferred, except for applications with very tight memory and/or high performance requirements.

2. Configuration Factory

The configuration factory allows to create configurations.

In most of the frameworks, the process of creating anything is extremely complicated and confusing, so the way the configuration factory creates the configuration is simplified to the maximum extent possible:

```
URL cfURL = new URL(...);
ConfigurationFactory cff = new ConfigurationFactory();
Configuration cf = cff.getConfiguration(cfURL);
```

That's all it takes. The configuration factory determines the type of the configuration object at that URL (plaintext or XML) and instantiates the corresponding configuration object.

The configuration URL protocols supported at this time are:

- `file:` file suffix is used to determine the configuration type, for .conf it is assumed to be plaintext, for .conf.xml it is assumed to be XML.
- `http:` MIME content type is used to determine the configuration type, for text/plain it is assumed to be plaintext, for text/xml it is assumed to be XML.
- `ftp:` same as for HTTP.

Support for other protocols is possible, especially if they provide data stream (unlike, for example, LDAP), but at this point creating such a driver was not a priority.

As of moment of writing, the configuration factory is also able to store the configuration, to a limited extent. Only `file:` protocol for plaintext is supported.

3. Configuration Chaining

Copyright © 1990 Vadim Tkachenko All rights reserved.

Configurations can be chained. The original idea is taken from the UNIX way of configuring the programs: there's a system-wide configuration, then goes the user default, then local configuration. Consider an example:

```
ConfigurationFactory cff;
Configuration systemCf = cff.getConfiguration(systemURL);
Configuration userCf = cff.getConfiguration(userURL);
Configuration localCf = cff.getConfiguration(localURL);

userCf.setDefaultConfiguration(systemCf);
localCf.setDefaultConfiguration(userCf);
```

In this case, the value will be read from `localCf`. If it is not present in the local configuration, it will be read from `userCf`, and if it is not present there, it will be read from `systemCf`.

4. Configurable Objects

For now, one word: `@ConfigurableProperty`. The rest is obvious.

See also: Instrumentation.

Copyright © 1990 Vadim Tkachenko All rights reserved.