A Simple URL Rewriter – TWEAK

Minor Project

IBM Career Education

Disclaimer

This Software Requirements Specification document is a guideline. The document details all the high level requirements. The document also describes the broad scope of the project. While developing the solution if the developer has a valid point to add more details being within the scope specified then it can be accommodated after consultation with IBM designated Mentor.

INTRODUCTION

The purpose of this document is to define scope and requirements of a simple URL Rewriter (TWEAK) for the developers of a software services company, engaged in developing Java Servlet/JSP based web applications. The proposed system will a mechanism, independent of web application server, to rewrite URLs for the purpose of making application SEO friendly and/or RESTful.

This document is the primary input to the development team to architect a solution for this project.

System Users

The entire team of developers of the software services company is expected to benefit from the simple URL rewriter, TWEAK.

Assumptions

1. For testing purpose, a simple servlet based test bed will be created to display the rewritten URLs.
2. It is assumed that the standard Java EE Filter Mechanism will be used for the development.

REQUIREMENTS

TWEAK will provide a mechanism, independent of web application server, to rewrite URLs. It will leverage the Java EE standard "filters" to intercept the URI and use regex to parse the same so that it can be rewritten. The following example illustrates the use of regex for parsing.

```
Incoming URL: http://www.example.com/products/9/
URI Component: products/9/
REGEX for matching: ^products\/(\d+)\/$
Substitution: http://www.example.com/show_prod.jsp?id=$1
```

This will result in to a single matched group with a value of "9". The substitution has only one parameter – "$1", where the value of first matched group will be placed. This will result in to the rewritten URL as "http://www.example.com/show_prod.jsp?id=9" URL.

The overall flow is outlined below:

1. Filter intercepts and captures the URI;
2. It is matched with the list of REGEX for pattern matching;
3. For the first matching pattern, all matching groups' values are extracted and new URI is formed by replacing parameters – first matching group to replace "$1", second matching group to replace ""$2", and so on. In event of no match, the URL is not changed.

4. Using either ServletRequest#getRequestDispatcher() and then RequestDispatcher#forward() to forward the request/response to the new URL (this will be a server-side redirect, not reflected in browser address bar), or cast the incoming ServletResponse to HttpServletResponse and then HttpServletResponse#sendRedirect() to redirect the response to the new URL (this will be a client side redirect, reflected in browser address bar).

Therefore, the TWEAK framework will consists of 2 key artifacts viz. (a) a filter to intercept requests for URL rewriting, and (b) a database table containing all the REGEX for pattern matching & extracting all matching group values.

The administrator of TWEAK will enter the REGEX using a properly designed form in to the database.

Optional Requirements

Consider developing a proper strategy for to handle multiple matches instead of stopping at the first match.

DEVELOPMENT ENVIRONMENT

TWEAK will be developed as a web application using Java/JSP/Servlets/Filters and DB2 database. Eclipse will be used as the IDE for the same. You may consider using a JavaScript framework like jQuery/Prototype/ Scriptaculous.

Details about Java EE 6 Filters can be found at http://docs.oracle.com/javaee/6/tutorial/doc/bnagb.html URL.