r4 - 07 May 2008 - 18:13:59 - RichardKettlewell

DisOrder Macro Language Changes

- DisOrder Macro Language Changes
  - New Syntax
  - Stored Procedures
  - Library Implementation
  - Existing Templates

DisOrder's macro language is rather hard to use. These changes are therefore proposed:

- The current syntax is too horrible to live. So it will be adapted to something easier for humans to get right.
- Stored procedures. There is a great deal of repetition. It could be eliminated by allowing new expansions to be defined within the templates.

Also the implementation has generic template expansion code mixed willy-nilly into DisOrder-specific code, making maintenance and further development harder. Separating the generic code into a library module is a convenient thing to do as part of the same exercise.

This is a major change so it'll probably be called DisOrder 4.0.

New Syntax

Expansions will take the form `@NAME{VALUE}{VALUE}...`.

Whitespace between arguments (outside the brackets) is permitted. The argument list continues until something that cannot be an argument is found.

Any of `{}`, `[]`, `()` can be used but the same ones must be used throughout. We don't allow `<>` because HTML tags are too likely to cause confusion.

`@@` will stand for a single `@`.

`@#` will start a comment that extends to the end of the line and *includes* the newline (so no blank line appears in the output; like M4's `dnl`.)

Stored Procedures

We'll add a new expansion, `define`. This will have three arguments:

1. The name of the macro to define
2. The list of argument names, separated by spaces
3. The expansion text

When macro is used as an expansion it must have exactly as many parameters as there were argument names. They will be assigned in the obvious way. The macro invocation is then expanded to the argument text, with the arguments acting as expansions.

Argument expansion will be lexical in scope.

For example:

```
@define {twice} {a}
        {@a @a}@#
@twice{xyzzy}@#
```

```
@twice{foo bar}
```

will expand to:

```
xyzzy xyzzyfoo bar foo bar
```

Library Implementation

The template expansion logic will be moved to `lib/`. Entry points will be:

- parse a template. This does a full recursive parse. Parse trees will be immutable.
- register expansions. The same table will be used for built-in expansions, simple expansions, magic expansions and macros.
- expand to a sink.

Simple expansions are those that take string parameters; magic expansions take parse trees.

When a macro is expanded, its parse tree is copied, with parameter values substituted in. The parameter values are not themselves expanded in the same way. The resulting tree is then expanded in the normal way.

Non-domain-specific expansions will be provided in a built-in set. The DisOrder-specific ones will still reside in the CGI.

Existing Templates

The existing templates will be rewritten using the new language.