MacroIssues

Macro bug fixes / enhancements

(edit: now fixed in svn head for Velocity 1.6. leaving this note up until Velocity 1.6 is released. Will)

some notes by Will <wglass@forio.com>.

I'm starting to look into the remain bug fix/enhancements for the 1.5 release.
Not surprisingly, many of the remaining issues are all pretty subtle. I thought I'd publically share a few notes on macros.

Here's a catalog of the open macro-related issues targeted for 1.5.

- http://issues.apache.org/jira/browse/VELOCITY-277 Macros in #parsed files are not refreshed when including page is refreshed
- http://issues.apache.org/jira/browse/VELOCITY-362 Can't load macros in file loaded with #parse
- http://issues.apache.org/jira/browse/VELOCITY-146 Macros not evaluated on the first attempt with #parse
- http://issues.apache.org/jira/browse/VELOCITY-24 Calls to local macros not always made when template caching is off
- http://issues.apache.org/jira/browse/VELOCITY-82 VM libs will not autoreload if unparseable at Velocity startup

—

Most of these issues stem from how Velocity stores macros. Macros can be stored either at a global level or on a per-template level. This is controlled by the config key "velocimacro.permissions.allow.inline.local.scope", which is false by default. (i.e. all macros are part of global scope). See the developer's guide.

This means that the following common-sense use case (referenced in VELOCITY-277, VELOCITY-362, and VELOCITY-136) just doesn't work.

file1.vm

```
#parse("macros.txt")

#NewMacro()
```

macros.txt

```
#macro(NewMacro) do something #end
```

Here's why. Case 1. If "velocimacro.permissions.allow.inline.local.scope" is true, then #NewMacro is defined in "macros.txt" but not "file1.vm". Thus the macro is not available for use.

Case 2. If "velocimacro.permissions.allow.inline.local.scope" is false, then then #NewMacro is defined globally and available from all templates. This seems like a good idea but is impractical in practice for the following reasons.

- First of all, it doesn't work. Hence the first three bugs. They are technically not bugs since this behavior is documented, though that's difficult to find.
- Currently macros are evaluated at parse time while #parse is evaluated at runtime. (allowing the template to dynamically choose a page to include). That's why the macro is not picked up by the #parse.
- Even if we fix this, there's a threading problem and a namespace problem. This macro is available to any template. If a website has hundreds or thousands of templates, there's no guarantee that another page hasn't defined a macro with the same name which could be loaded into the same common name space unexpectedly. The results could be very hard to debug.

Proposed Solution: Add a new mode for macros. (really, this should be the default in the future), which is request based. When a macro is defined in a page, it should be follow the execution of the page, be available in any subpages called with #parse and be available to a calling page that has called this with #parse. However, the macro shouldn't interfere with the global namespace or any non-related templates.

Open issues / discussion points -

- Can this be done while preserving Velocity's high-performance template caching?
- If we add a new scope for macros, is this backwards compatible? (assuming we allow old modes to continue to work).