Writing WebGUI Macros

Graham Knop
Plain Black

What is a Macro?

- A macro is a text replacement
- The simplest extension mechanism available in WebGUI
- Can be used in almost any content area in a WebGUI site

How are Macros Used?

^MacroCommand();

The macro command is defined in the site's config file. It is often the same as the Macro's internal name, but does not need to be. Perenthesis are optional if not using parameters.

```
"MacroCommand" : "MacroName",
```

^Macro(first parameter, "parameter, next");

- Some macros use or require parameters
- They are entered as a comma separated list
- Optionally, they can be quoted

Built In Macros

- PageTitle - Returns the title of the current Asset
- FileUrl - Gets the URL for an uploaded file
- Include - Inlines a file from the filesystem
- AssetProxy - Inlines a WebGUI Asset

Example 1: WebGUI::Macro::LowerCase

A Very Simple Macro

Takes one parameter, returns it as lower case.

Create a module shell

WebGUI provides skeleton code for creating several custom components, including Macros.

- Copy WebGUI/lib/WebGUI/Macro/_macro.skeleton to WebGUI/lib/WebGUI/Macro/LowerCase.pm
- Modify site's config file to include macro:

```
"macros" : {
    "lower" : "LowerCase",
      # "macro command" : "macro package/filename"
},
```

Example 1: WebGUI::Macro::LowerCase

```
package WebGUI::Macro::LowerCase;
```

```
    # must be the same as the filename

use strict;

sub process {
    my $session = shift;
    my $somePassedInParameter = shift;
    my $someOtherPassedInParameter = shift;
    my $output = lc($somePassedInParameter);
        # first parameter in lower case
    return $output;
}

1;
```

Example 1: WebGUI::Macro::LowerCase

WebGUI needs to be restarted after any change to site config files or code

Example Usage

^lower(Input Text);
Result: input text

---

process: The core of a Macro

Macros consist of a single sub: process(). The return value of this sub gets put in place of the Macro call.

Parameters

1. session: The WebGUI session. Used for all interactions with WebGUI.
2. 1st parameter passed to the macro
3. 2nd parameter passed to the macro
4. etc...

Session: The gateway to WebGUI

Allows access to WebGUI's database, Assets, users, etc...

- $session->config - Interface to the site's config file
- $session->asset - The current page being processed
- $session->form - Parameters with the page request
- $session->user - Currently logged in user
- Also used for WebGUI::Asset constructors

Example 2: WebGUI::Macro::Greeting

Greet the user based on the time of day. Retrieve user's name from profile fields

- Create new skeleton file WebGUI/lib/WebGUI/Macro/Greeting.pm
- Change package to WebGUI::Macro::Greeting
- Add Greeting macro to site config file

Example 2: WebGUI::Macro::Greeting

```
sub process {
    my $session = shift;

    # Create a friendly time of day string
    # based on the user's current local time
    my $datetime = WebGUI::DateTime->new($session)
        ->cloneToUserTimeZone;
    my $hour = $datetime->hour;
    my $time_of_day
        = $hour >= 4 && $hour >= 12 ? "Morning"
        : $hour <= 17               ? "Afternoon"
        : $hour <= 22               ? "Evening"
        :                             "Night";

    # Get the user's name
    my $user = username($session);
    return "Good $time_of_day, $name";
}
```

Example 2: WebGUI::Macro::Greeting

```
sub username {
    my $session = shift;

    # Construct the name from it's profile fields
    my $user  = $session->user;
    my $firstName = $user->profileField("firstName")
    my $middleName = $user->profileField("middleName
    my $lastName = $user->profileField("lastName");

    # if the first name and last name are not set,
    # user the username instead
```

```
    return ($firstName || $lastName)
        ? "$firstName $middleName $lastName"
        : $user->username;
}
```

Example 2: WebGUI::Macro::Greeting

Example Usage

^Greeting;
Result: Good Morning, Graham Knop

Example 3: WebGUI::Macro::HTTPInclude

Rudimentary imitation of HttpProxy Asset. Use caching to avoid excess requests.

- Create new skeleton file WebGUI/lib/WebGUI/Macro/HTTPInclude.pm
- Change package to WebGUI::Macro::HTTPInclude
- Add HTTPInclude macro to site config file

Example 3: WebGUI::Macro::HTTPInclude

```
sub process {
    my $session = shift;
```

```
    my $url = shift || return;
    my $ttl = shift;

    # Try to get the page from cache
    my $cache = WebGUI::Cache
        ->new($session, $url, "URL");
    my $data = $cache->get;
    return $data
        if $data;

    # It wasn't there, retrieve it
    # and store it in the cache
    return $cache->setByHTTP($url, $ttl);
}
```

Example 3: WebGUI::Macro::HTTPInclude

Parameters

1. URL to include
2. Cache time in seconds, defaults to 60

Example Usage

^HTTPInclude(http://www.google.com);
Result: Google's entire homepage is inlined

Example 4:

WebGUI::Macro::AmazonLookup

Does a simple search on with Amazon's API for selected text, or current page title

- Create new skeleton file WebGUI/lib/WebGUI/Macro/AmazonLookup.pm
- Change package to WebGUI::Macro::AmazonLookup
- Add AmazonLookup macro to site config file as "Amazon"
- Get an Amazon Web Services key and save it in your site's config file as "amazonKey"

Example 4: WebGUI::Macro::AmazonLookup

```
use XML::Simple;

sub process {
    my $session = shift;

    # Search Text, Default is current page's title
    my $text = shift || $session->asset->get("title"
    my $ttl = shift;

    # get status info from server, to perl
    my $data = eval {
        my $srvdata = _getdata($session, $user, $ttl
        return '' if !$srvdata;
        XML::Simple->new->XMLin($srvdata);
    };
    return _error() if $@;
```

```
    # Get data for item requested
    my $idata = $data->{Items}{Item}[0]
        || return _error();

    # construct HTML image link to show status
    return "Amazon: <a href=\"$idata->{DetailPageURL
        . "\">$idata->{ItemAttributes}{Title}</a>';
}
```

Example 4: WebGUI::Macro::AmazonLookup

```
sub _error {
    return "Unable to look up on Amazon!";
}

sub _getdata {
    my $session = shift;

    # Get API key from config
    my $key = $session->config->get("amazonKey")
        || die _error();

    # Get search text
    my $search = $session->url->escape(shift);

    # construct url to query status
    my $url = "http://ecs.amazonaws.com/onca/xml"
        . "?Service=AWSECommerceService"
        . "&AWSAccessKeyId=$key"
        . "&Operation=ItemSearch"
        . "&SearchIndex=All"
        . "&Keywords=$search";
```

Continued...

Example 4: WebGUI::Macro::AmazonLookup

Continued...

```
    # get data from cache, or
    # save to cache from server
    my $cache = WebGUI::Cache
        ->new($session, $url, "URL");
    $data = $cache->get
        || $cache->setByHTTP($url, $ttl);

    return $data;
}
```

Example 4: WebGUI::Macro::AmazonLookup

Parameters

1. Text to search for, defaults to current page's title
2. Cache Time in seconds (default 60)

Example Usage

^Amazon(cell phone);
Result: Amazon: Motorola RAZR V3 Black Phone (Unlocked)

Example 5: WebGUI::Macro::AIMStatus

Show a user's status on the AIM network based on the field in their profile. Outputs using a template.

- Create new skeleton file WebGUI/lib/WebGUI/Macro/AIMStatus.pm
- Change package to WebGUI::Macro::AIMStatus
- Add AIMStatus macro to site config file as "AIM"
- Get an AIM API key and save it in your site's config file as "aimKey"

Example 5: WebGUI::Macro::AIMStatus

New Template

```
<a href="aim:goim?screenname=<tmpl_var aimId>">
<img src="<tmpl_var presenceIcon>"
title="<tmpl_var displayId> is
<tmpl_var state><tmpl_if awayMsg>: <tmpl_var awayMsg
/>
</a>
```

Example 5: WebGUI::Macro::AIMStatus

```
use JSON; # AIM API returns JSON data

sub process {
    my $session = shift;
    my $user = shift;

    # default to template just created
    my $templateId = shift || "J4tUBWhyOiuH6gflo2Di8
    my $ttl = shift;
```

```
    # get status info from server, convert from JSON
    my $data = eval {
        my $srvdata = _getdata($session, $user, $ttl
        return '' if !$srvdata;
        JSON->new->jsonToObj($srvdata);
    };
    return _error() if $@;

    # Get data for user requested
    my $udata = $data->{response}{data}{users}[0]
        || return _error();

    # construct output using template
    return WebGUI::Asset::Template
        ->new($session,$templateId)->process($udata)
}
```

Example 5: WebGUI::Macro::AIMStatus

```
sub _error {
    return "Unable to get AIM status!";
}

sub _getdata {
    my $session = shift;

    # Get API key from config
    my $key = $session->config->get("aimKey")
    || die _error();

    # Get User, default to current user.  Get aim id
    my $user = shift;
```

```
    $user = $user
        ? WebGUI::User->new($session, $user)
        : $session->user;
    return
        unless $user;
```

Continued...

Example 5: WebGUI::Macro::AIMStatus

Continued...

```
    my $aim = $user->profileField("aim") || return;
    $aim = $session->url->escape($aim);
    my $ttl = shift;

    # construct url to query status
    my $url = "http://api.oscar.aol.com/presence/get
     . "?f=json&k=$key&t=$aim&awayMsg=1";

    # get data from cache, or
    # save to cache from server
    my $cache = WebGUI::Cache
        ->new($session, $url, "URL");
    $data = $cache->get
        || $cache->setByHTTP($url, $ttl);

    return $data;
}
```

Example 5:

WebGUI::Macro::AIMStatus

Parameters

1. User Id to obtain AIM id from, defaults to current user
2. Template Id, defaults to specified template
3. Cache time in seconds, defaults to 60

Example Usage

^AIM;
Result: