Template Toolkit

Shlomi Fish
YAPC::Israel::2004

Case Study

- Suppose you want to create a parameterized HTML fragment.
- How do you do it?

How to do it

```
sub template
{
    my %args = (@_);
    my $toolbox = $args{'toolbox'};
    my $extra_class = $toolbox ? " recadmin" : "";
    my $toolbox_div = "";
    my $id = $args{id};
    if ($toolbox)
    {
        $toolbox_div = "<div class=\"recordtoolbox\">\n<p>\n" .
            "<a href=\"./edit/?id=${id}\">Edit</a><br />\n" .
            ($args{enabled} ?
              "<a href=\"./disable/?id=${id}\">Disable</a><br />" :
              "<a href=\"./enable/?id=${id}\">Enable</a><br />"
            )  .
            "</p>\n</div>\n";
    }
    return "<div class=\"record$extra_class\">\n<h3>$args{title}</h3>\n" .
        "<div class=\"desc\">$args{description}</div>\n".
        "<p class=\"data\"><b>Phone:</b> $args{phone}<br />\n" .
        "<b>E-mail:</b> $args{email}<br /></p></div>\n" .
        $toolbox_div;
}
```

How to do it Right

- Use a Templating System like Template Toolkit
- Embed the fields and directives inside the code
- Separate Content from Display

Same Thing, but Better

```
<div class="record[% IF toolbox %] recadmin[% END %]">
<h3>[% title %]</h3>
<div class="desc">
[% description %]
</div>
<p class="data">
<b>Phone:</b> [% phone %]<br />
<b>E-mail:</b> [% email %]<br />
</p>
</div>
[% IF toolbox %]
<div class="recordtoolbox">
<p>
<a href="./edit/?id=[% id %]">Edit</a><br />
[% IF enabled %]
<a href="./disable/?id=[% id %]">Disable</a><br />
[% ELSE %]
<a href="./enable/?id=[% id %]">Enable</a><br />
[% END %]
</p>
</div>
[% END %]
```

What is the Template Toolkit?

- One of the most comprehensive template systems on CPAN.
- Lots of built-in functionality and add-ons.
- Interfaces easily with other modules.
- User-defined formatting escapes.
- Actively Developed (Template Toolkit 3 is in the works).

Why should you use it?

> “Any sufficiently complicated C or Fortran program contains an ad-hoc informally-specified bug-ridden slow implementation of half of Common Lisp”
>
> *Phil Greenspun*

Which means:

> “Any sufficiently complicated C or Fortran program contains an ad hoc informally-specified bug-ridden slow implementation of half of Common Lisp.”
>
> *Phil Greenspun*

- If you write an ad-hoc templating system, you are bound to encounter limitations after a while.
- Soon, your template system will have more and more features.
- At the same time, its code will become more and more ugly.
- So, use Template Toolkit – the “Common Lisp” of templating systems.

Speaking from Experience…

End