

HTML::Template

Chicago Perl Mongers
November 16, 2004

Why template?

- :: Keep your code neat

 - :: No HTML in code

 - :: Code can be reused with different templates

- :: Easier editing for designers

 - :: Can't break complex Perl syntax

 - :: Exposes simple HTML tags that designers are familiar with

The Choices

:: HTML::Template (H:T)

:: Template Toolkit (TT)

:: Mason

:: Many others

H:T Pros

:: Simple

:: Fast

:: Easy to use

:: Easy to implement

:: Pure separation of logic and look

H:T Cons

- :: Not terribly powerful
- :: Only handles HTML templates well, but can handle other formats

TT Pros

- :: Template just about everything
- :: Pretty darn fast
- :: Can define your own tag format

TT Cons

- :: Not as fast as H:T
- :: Not as simple to teach to designers as H:T
- :: Overkill for some environments

Mason Pros

- :: Phenomenal cosmic power
- :: Almost it's own programming language

Mason Cons

- :: Not really usable outside of mod_perl
- :: Big overhead
- :: Way slower than H:T & TT

WebGUI uses H:T

- :: Easy for users
- :: Super fast
- :: Caching
- :: Supporting SP sucks
- :: Separation of of logic and look



I Hate SP

Server Page Languages Suck!!!

Disclaimer: I'm biased

The API

```
use HTML::Template;
my $t = HTML::Template->new(
    filename => $filename,
    scalarref => \$template
);
$t->param(%vars);
print $t->output;
```

Variables

```
$t->param( username => "JT" );
```

```
<b>user:</b><tmpl_var username>
```

Conditions

```
$t->param( isAdmin => 1);
```

```
<tmpl_if isAdmin>
```

```
  This user is an admin.
```

```
</tmpl_if>
```

Conditions

```
$t->param(username => "");
```

```
<tmpl_if username>  
  Hello <tmpl_var username>.  
<tmpl_else>  
  Hello Visitor.  
</tmpl_if>
```


Loops

```
$t->param(people=>[{  
  name=>"JT",  
  email=>'jt@plainblack.com'  
},{  
  name=>"Andy",  
  email=>'andy@petdance.com'  
}]);
```


Loops

```
<tmpl_loop people>  
  <p>  
    Name: <tmpl_var name><br />  
    Email: <tmpl_var email>  
  </p>  
</tmpl_loop>
```

Loops of Loops

```
push(@months, {  
    days=>\@days,  
    monthname=> $name,  
});  
$t->params(months=>\@months);
```

Loops of Loops

```
<tmpl_loop months>  
  <tmpl_if days><p>  
    <tmpl_var monthname><br>  
    <tmpl_loop days> ...etc...  
  </tmpl_loop>  
</p></tmpl_if>  
</tmpl_loop>
```

Filters

```
my $coderef = sub {  
  my $text = shift;  
  $text =~ /Chicago/London/ig;  
};
```

```
my $t = HTML::Template->new(  
  filename=>$filename,  
  filter => $coderef  
);
```

Object Associations

```
my $cgi = CGI->new;
```

```
my $t = HTML::Template->new(  
  filename=>$filename,  
  associate=>$cgi  
);
```

Requires the object to have a `param()` method

H:T Caching

- :: Increase performance by up to 100%
- :: Memory - Both IPC and not
- :: File
- :: Mixed - File and Memory

H:T Tags

:: <tmpl_var *>

:: <tmpl_if *>

:: <tmpl_else>

:: <tmpl_unless *>

:: <tmpl_loop *>

Tag Syntax

```
<tmpl_var email>
```

```
<tmpl_var name="email">
```

```
<!-- tmpl_var name="email" -->
```


Tag Options

```
<tmpl_var name="color"  
  default="blue">
```

```
<tmpl_var name="description"  
  escape="html||url||js">
```

Embedded Tags

```
<input type="text"  
name="description"  
value="<tmpl_var  
name="description"  
escape="html">">
```

Loop Context Vars

```
<tmpl_loop pages>  
  <tmpl_if __first__></tmpl_if>  
  <tmpl_if __last__></tmpl_if>  
  <tmpl_if __inner__></tmpl_if>  
  <tmpl_if __odd__></tmpl_if>  
</tmpl_loop>
```

Add-ons

- :: Lots of them
- :: Every benefit comes with a detriment
- :: Usually speed
- :: Sometimes compatibility/limitations

H:T::Expr

:: Adds expressions

:: <tmpl_var EXPR="head_count >= 8">

:: Adds functions

:: <tmpl_var EXPR="sprintf("%3s",3.14)">

:: Define your own functions

H:T::JIT

- :: Compiles H:T templates to C code
- :: Up to 1000% performance increase
- :: Has many restrictions

H:T::HashWrapper

:: Blesses hash or reblesses object to work with H:T's associate option

:: Written by our own Greg Fast

H:T::Dumper

- :: API compatible with H:T
- :: Dumps template data with Data::Dumper
- :: Useful for testing
 - :: Replace **use H:T:D** with **use H:T** after testing

H:T:Set

:: Allows you to set arbitrary variables in the template

:: `<tmpl_set name="title">my title</tmpl_set>`

:: Exports environment variables as `ENV_*`

:: `<tmpl_var env_remote_addr>`

H:T::Associate

- :: Bridge between H:T and any object
- :: Creates profiles of object data to map to H:T's associate option

H:T::Associate:: FormValidator

:: Hooks up Data::FormValidator::
Results with H:T's associate option

H:T::Filter:: Dreamweaver

- :: Converts Dreamweaver template files (.dwt) into H:T templates
- :: Requires H:T:Expr

H:T::XPath

- :: Easily template XML data
- :: Retrieves data using XPath

H:T::Associate:: FormField

:: Easily generate forms using H:T

Template::Plugin::H:T

:: Use H:T templates in Template Toolkit

CGI::Application

:: A CGI app framework that has native support for H:T

More Information

<http://html-template.sf.net>