

Template::Zoom - Modern HTML and PDF Engine

Stefan Hornburg (Racke)
racke@linuxia.de

Perl Oasis 2011, 15th January 2011

Why

- ▶ Separation of web design and programming
- ▶ How available template engines violate this principle
 - ▶ Mini language (Template::Toolkit)
 - ▶ Inline code
 - ▶ CSS selectors (HTML::Zoom)
- ▶ Solutions by Template::Zoom
 - ▶ Static HTML file
 - ▶ Specification file
- ▶ Further Goals
 - ▶ Great flexibility
 - ▶ Tweaks through tree manipulations

Cart: Hash

```
$cart = [  
  {isbn => '978-0-2016-1622-4',  
   title => 'The Pragmatic Programmer',  
   quantity => 1, price => 49.95},  
  
  {isbn => '978-1-4302-1833-3',  
   title => 'Pro Git',  
   quantity => 1, price => 34.99},  
];
```

Cart: HTML Template

```

<table class="cart">
<tr class="carthead">
<th>Name</th><th>Quantity </th><th>Price </th>
</tr>
<tr class="cartitem">
<td class="name">Perl 6</td>
<td><input class="quantity" name="quantity" size="3" value="10"></td>
<td class="price">$$$</td>
</tr>
<tr class="carthead">
<th colspan="2"></th><th>Total </th></tr>
<tr>
<td colspan="2"></td><td class="cost">$$$</td></tr>
</table>

```

Cart: ITL

```

<table class="cart">
<tr class="carthead">
<th>Name</th><th>Quantity </th><th>Price </th>
</tr>
[ item-list ]
<tr class="cartitem">
<td class="name">[item-modifier title]</td>
<td><input class="quantity" name="quantity" size="3" value="[item-qua
<td class="price">[item-price]</td>
</tr>
[/item-list ]
<tr class="carthead">
<th colspan="2"></th><th>Total </th></tr>
<tr>
<td colspan="2"></td><td class="cost">[total-cost]</td>
</tr>

```

Cart: Template::Toolkit

```
<table class="cart">
<tr class="carthead">
<th>Name</th>
<th>Quantity</th>
<th>Price</th>
</tr>
[% FOREACH cart %]
<tr class="cartitem">
<td class="name">[% title %]</td>
<td><input class="quantity" name="quantity" size="3" value="[% quanti
<td class="price">[% price %]</td>
</tr>
[% END %]
</table>
```

Cart: HTML::Zoom

```
use HTML::Zoom;
```

```
$cart = ...
```

```
$zoom = HTML::Zoom->from_file('cart.html');
```

```
$zoom->select('.cartitem')->repeat_content([
```

```
  map { my $field = $_; sub {
```

```
    $_[0]->select('.name')->replace_content($field->{title});
```

```
    $_[0]->select('.quantity')->replace_content($field->{quantity});
```

```
    $_[0]->select('.price')->replace_content($field->{price});
```

```
  };
```

```
  ] @$cart]);
```

```
print $zoom->to_html();
```

Template Problems

- ▶ Mini language in HTML template
- ▶ Dynamic pages (border cases, errors, ...)

Template::Zoom Concept

- ▶ Specification
- ▶ Template
- ▶ Data or objects (iterator)

Template::Zoom Specification (XML)

```
<specification name="cart" description="Cart">  
<list name="cart" class="cartitem" iterator="cart">  
<param name="name" field="title" />  
<param name="quantity" />  
<param name="price" />  
</list >  
<value name="cost" />  
</specification >
```

Template::Zoom Specification (Config::Scoped)

```
list cart {
  class = cartitem
  iterator = cart
}
param quantity {
  list = cart
}
param price {
  list = cart
}
param name {
  list = cart
  field = title
}
value cost {
  name = cost
```

Template::Zoom Script (XML)

```
use Template::Zoom;

my ($cart, $zoom, %values);

$cart = ...
$values{cost} = ...

$zoom = new Template::Zoom(specification_file => 'cart.xml',
                             template_file => 'cart.html',
                             iterators => {cart => $cart},
                             values => \%values,
);

print $zoom->process();
```

Template::Zoom Script (Config::Scoped)

```
use Template::Zoom;

my ($cart, $zoom, %values);

$cart = ...
$values{cost} = ...

$zoom = new Template::Zoom( specification_file => 'cart.conf',
                             specification_parser => 'Scoped',
                             template_file => 'cart.html',
                             iterators => { cart => $cart },
                             values => \%values,
);

print $zoom->process ();
```

Menus: Database table

```
CREATE TABLE menus (  
  code int NOT NULL auto_increment,  
  name varchar(255) NOT NULL DEFAULT '',  
  url varchar(255) NOT NULL DEFAULT '',  
  menu_name varchar(64) NOT NULL DEFAULT '',  
  permission varchar(64) NOT NULL DEFAULT '',  
  weight int NOT NULL DEFAULT 0,  
  PRIMARY KEY(code),  
  KEY(menu_name)  
);
```

Menus: Specification

```
<specification name="menu" description="Menu">
<list name="menu" class="menu" table="menus">
<input name="name" required="1" field="menu_name"/>
<param name="label" field="name"/>
<param name="url"/>
</list >
</specification >
```

Menus: Template

```
<ul class="menu">  
<li><a href="" class="url"><span class="label"></span></li >  
</ul>
```


Menus: Script

```
use Template::Zoom;  
use Template::Zoom::Database::Rose;  
  
$db_object = new Template::Zoom::Database::Rose(dbname => 'temzoo',  
        dbtype => 'Pg',  
);  
  
$zoom = new Template::Zoom(specification_file => 'menu.xml',  
        template_file => 'menu.html',  
        database => $db_object,  
);  
  
$zoom->process({name => main});
```

Iterators

- ▶ next method
- ▶ count method
- ▶ hash as return value
- ▶ Template::Zoom::Iterator

Lists with alternating rows

```
<table class="cart">
<tr class="carthead">
<th>Name</th><th>Quantity </th><th>Price </th>
</tr>
<tr class="cartitem odd">
<td class="name">Perl 6</td>
<td><input class="quantity" name="quantity" size="3" value="10"></td>
<td class="price">$$$</td>
</tr>
<tr class="cartitem even">
<td class="name">Pro Git</td>
<td><input class="quantity" name="quantity" size="3" value="10"></td>
<td class="price">$$$</td>
</tr>
</table>
```

Filter: Specification

```
<specification name="menu" description="Menu">
<list name="menu" class="menu" table="menus">
<input name="name" required="1" field="menu_name"/>
<param name="label" field="name"/>
<param name="url" target="href" filter="link"/>
</list >
</specification >
```

Filter: Function

```
sub link_filter {  
  my $page = shift;  
  my $url;  
  
  $url = ...  
  
  return $url;  
}
```

```
$zoom = new Template::Zoom(specification_file => 'menu.xml',  
  template_file => 'menu.html',  
  database => $db_object,  
  filters => {link => \&link_filter},  
);
```

Global Filter

```
<specification name="menu" description="Menu">
<filter name="acl" field="permission"/>
<list name="menu" class="menu" table="menus">
<input name="name" required="1" field="menu_name"/>
<param name="label" field="name"/>
<param name="url" target="href" filter="link"/>
</list >
</specification >
```

Sort: Specification

```
<specification name="menu" description="Menu">
<list name="menu" class="menu" table="menus">
<sort name="default">
<field name="weight" direction="desc"/>
<field name="code" direction="asc"/>
</sort>
<input name="name" required="1" field="menu_name"/>
<param name="label" field="name"/>
<param name="url" target="href" filter="link"/>
</list >
</specification >
```

I18N

```
sub translate {  
    my $text = shift;  
  
    ...  
  
    return $text;  
}
```

```
$i18n = new Template::Zoom::I18N (\&translate);
```

```
$zoom = new Template::Zoom(specification_file => 'menu.xml',  
    template_file => 'menu.html',  
    database => $db_object,  
    i18n => $i18n,  
);
```


I18N: Lookup Keys

```
<i18n name="returnurl" key="RETURN_URL"/>
```

Forms: Specification

```
<specification name='search' description=''>  
<form name='search'>  
<field name='searchterm' />  
<field name='searchsubmit' />  
</form>  
</specification >
```

Forms: Manipulating

`set_action` Changing form action

`set_method` Changing form method (GET, POST)

`fill` Fill form fields

HTML to PDF

- ▶ HTML template processing
- ▶ PDF conversion (PDF::API2)
 1. calculate
 2. partition
 3. render
- ▶ Inline CSS

PDF: Code

```
$zoom = new Template::Zoom ( specification_file => 'pdf.xml',  
                             template_file => 'pdf.html',  
                             values => \%values );  
  
$zoom->process ();  
  
$pdf = new Template::Zoom::PDF ( template => $zoom->template (),  
                                 file => 'invoice.pdf' );  
  
$pdf->process ();
```

PDF: Import

```
$import{file} = 'shippinglabel.pdf';  
$import{scale} = 0.8;  
$import{margin} = {left => '3mm', top => '6mm'};  
  
$pdf = new Template::Zoom::PDF (template => $zoom->template(),  
                                file => 'invoice.pdf',  
                                import => \%import);
```

Current and Future Use Cases

- ▶ Very Large Product Lists
- ▶ Shop Backend
 - ▶ Product Editor
 - ▶ Product Search & Replace
- ▶ PDF Invoices
- ▶ Template Engine for Interchange

Roadmap

- ▶ Documentation
- ▶ Tests
- ▶ Conditions
- ▶ Empty lists, number of results
- ▶ Selected items
- ▶ Paging
- ▶ Trees

The End

Git <http://git.linuxia.de/?p=temzoo.git;a=summary>
[git://git.linuxia.de/temzoo.git](http://git.linuxia.de/temzoo.git)

CPAN not yet

Website not yet

Release tomorrow?

Talk <http://www.linuxia.de/talks/opw2011/temzoo-opw2011-beamer.pdf>

Questions ???