```
package ZTab;
$VERSION = '0.04';
use strict;

{
    # The Package variables

    my $_tab_counter = 1;
    sub get_tab_counter   { $_tab_counter; }
    sub _incr_tab_counter { $_tab_counter++; }
}

my $css = {
           body =>      {iname => 'body', # not used !
                         margin=> 0,
                         padding =>  0,
                         'background-color' => '#e3e6e9',
                         height =>  '100%',
                        },
           content =>   {name => '.content',
                         position => 'relative',
                         'margin-left' => '0%',
                         top =>  '0px',
                         'font-family' =>  'Trebuchet MS, Arial, sans-serif',
                         padding =>  '8px 12px 12px 12px',
                         border =>  '1px solid #666',
                         width =>  '100%',
                         'border-top' =>  '1px solid #999',
                         'border-left' =>  '1px solid #666',
                         'z-index' =>  500,
                        },
           div_tabs =>  {name => 'div.tabs',
                         position => 'relative',
                         'font-size' => '12px',
                         'line-height' => '15px',
                         top =>  '0px',
                         left => '0%',
                         'white-space' =>  'nowrap',
                         'font-family' =>  'Verdana, Arial, sans-serif',
                         'cursor' =>  'default !important',
                         'font-weight' =>  '700 !important',
                         'z-index' =>  10000,
                        },
           tab =>       {name => '.tab',
                         border => '1px solid #347',
                         padding => '2px 9px 1px 9px',
                         'background-color' => '#fff000',
                         'z-index' => 100,
                         'border-bottom-width' => 1,
                        },

           tabHover => {name => '.tabHover',
                        border => '1px solid #347',
                        padding => '2px 9px 1px 9px',
                        'background-color' => '#fff000',
                        'z-index' =>  100,
                        'border-bottom-width' => 1,
                        },

           tabActive => {name => '.tabActive',
                         border => '1px solid #347',
                         padding => '2px 9px 1px 9px',
                         'background-color' =>  '#ffffff',
                         'z-index' => 100,
                         'border-bottom-width' => 1,
                         },

           div_tabsClone => {name => 'div.tabsClone',
                             'border-top-width' => '0 !important',
                             'border-bottom-width' => '1px !important',
                            },

           removeTab =>  {name => 'a.removeTab',
                          'font-size' => 'smaller',
                          'text-decoration' =>  'none',
                         },

           TabParams =>  {
                          useClone => 'false',
                          alwaysShowClone => 'false',
                          eventType =>  '"click"',
                          tabTagName =>  '"span"',
                         },
};

# BODY section

sub body_attr { ' onload="tabInit()"' }

# CSS section

sub make_css (%) {
 my %parm = @_;
 local $_;

 foreach my $csskey ( keys %parm ) {
   map {
         $css->{$csskey}->{$_} = ${$parm{$csskey}}{$_}
       } (keys %{$parm{$csskey}})
   if ( exists $css->{$csskey} ); # check only valid keys (css)
 }
 my $CSS = "<style type=\"text/css\">\n<!--\n";;
 foreach my $csskey ( keys %{$css} ) {
      next if ( !exists $css->{$csskey}->{name} ); # if not in css section
      $CSS .= $css->{$csskey}->{name} . "{\n";
      foreach my $key ( keys %{$css->{$csskey}} ) {
          $CSS .= $key . ':'. $css->{$csskey}->{$key} . ";\n" if ( $key ne 'name' );
      }
      $CSS .= '}'."\n";
 }
 $CSS .= "//-->\n</style>\n";
 return ( $CSS );
}


# JS section

sub make_js (%) {
 my %parm = @_;
 local $_;

 my $jslib  = $parm{'jslib'};
 my $base   = $parm{'base'};
```

```
my @tmp=();
map { push @tmp, $_ . ':' . $css->{TabParams}->{$_} } (keys %{$css->{TabParams}});
my $TabParams = join(",",@tmp);
my $JS ="";
$JS .="<BASE = \"$base\">\n" if $base =~/file:\/\//i;
$JS .=<<EOT;
<script src="$jslib/utils.js" type="text/javascript"></script>
<script src="$jslib/viewport.js" type="text/javascript"></script>
<script src="$jslib/global.js" type="text/javascript"></script>
<script src="$jslib/cookie.js" type="text/javascript"></script>
<script src="$jslib/tabs.js" type="text/javascript"></script>
<script type="text/javascript">
TabParams = { $TabParams };
</script>
EOT
 return ( $JS );
}

# Methods

sub new {
    my $class = $_[0];
    my $id = $class->get_tab_counter();
    my $label = $id . '_' if($id);

    my $objref = {
        _id       => $id, # unical object ID
        _label    => $label,         # unique label for names (not used now)
        _tabs     => [],
        _contents => [],
    };

    $class->_incr_tab_counter();

    bless  $objref, $class;
}

sub get_label { $_[0]->{_label} }

sub last_tab {
    my ($self) = @_;
    my $len1 = $#{$self->{_tabs}};
    my $len2 = $#{$self->{_contents}};
    return (($len1 > $len2) ? $len1 : $len2 );
}

sub add_tab {
    my ($self,%arg) = @_;
    my $new_last = $self->last_tab() + 1;

    $self->{_tabs}->[$new_last] = $arg{'tab'};
    $self->{_contents}->[$new_last] = $arg{'content'};
}

sub set_tab {
    my ($self,$index,%arg) = @_;

    $self->{_tabs}->[$index] = $arg{'tab'};
    $self->{_contents}->[$index] = $arg{'content'};
}

sub get_tab {
    my ($self,%arg) = @_;
    my $active = $arg{'active'} || 1;
    my $label   = $self->get_label();
    my $span =  $css->{TabParams}->{tabTagName};
    $span =~s/"//g;
    my $out = '<div class="tabs">' . "\n";
    for(my $i=0;$i<=$self->last_tab();$i++) {
        if( defined($self->{_tabs}->[$i]) || defined($self->{_contents}->[$i]) ) {
            my $class = ($i+1==$active) ? 'tab tabActive' : 'tab';
            $out .= '<'.$span.' id="tab' . $label . ($i+1) . '" class="' .
                $class . '">' . $self->{_tabs}->[$i] . '</'.$span.'>' . "\n";
        }
    }
    $out .= "</div>\n\n";

    return($out);
}

sub get_content {
    my ($self) = @_;
    my $label   = $self->get_label();

    my $out;
    for(my $i=0;$i<=$self->last_tab();$i++) {
        if( defined($self->{_tabs}->[$i]) ||
            defined($self->{_contents}->[$i]) )
        {
            $out .= '<div id="content' . $label . ($i+1) .
                '" class="content">' . $self->{_contents}->[$i] .
                '</div>' . "\n";
        }
    }

    return($out);
}


sub test {
print $css->{content}->{'margin-left'},"\n";
print $css->{tab}->{'background-color'},"\n";
}

1;
```