D.R.Y. Don‘t repeat yourself

FrOSCon 2017

Jan Büren

kivitendo GmbH

D.R.Y. Don‘t repeat yourself

Besser stumpfsinnig, aber konsequent **ein** Design-Pattern anwenden anstatt sich theoretisch im Kreis zu drehen.

Why D.R.Y. ?

- Redundanzen sind aufwändiger!
- Duplikate sind nicht 100%ig Duplikate
- D.R.Y. einmal anwenden → nur erweitern

Quelle: https://de.wikipedia.org/wiki/Don%E2%80%99t_repeat_yourself

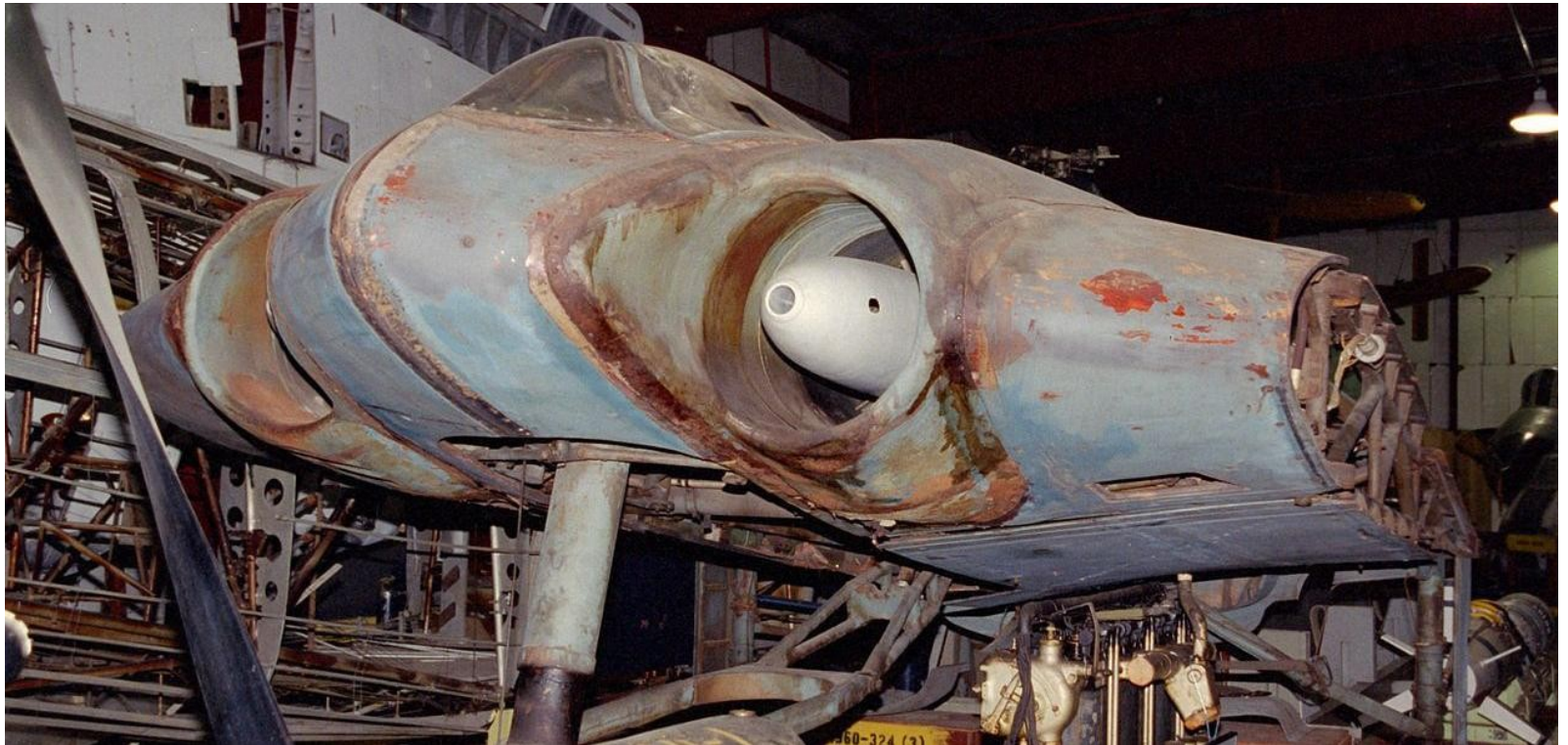
Mein Programmier-Einschätzung: 5 von 10

- Code simpel
- Code erweiterbar
- Code verständlich
- Code defensiv
- Idiotensicher

Die erste Version ist nicht die Beste

=> Prototypen an unkritischer Stelle starten ;-)

=> Testfälle automatisieren



```
[%- USE T8 %]
[%- USE HTML %]
<h1>[% HTML.escape(title) %]</h1>

 [%- IF message %]
 <p>
  [% HTML.escape(message) %]
 </p>
 [%- END %]

 <form method="post" action="generictranslations.pl" id="form">

  <table>

   <tr>
    <th class="listheading"> </th>
    <th class="listheading">[% 'Remittance information prefix' | $T8 %]</th>
   </tr>

   [%- FOREACH language = LANGUAGES %]
   <tr>
    <td>
     [%- IF language.id == 'default' %]
     [% 'Default (no language selected)' | $T8 %]
     [%- ELSE %]
     [%- HTML.escape(language.description) %]
     [%- END %]
    </td>
    <td><input name="translation__[% language.id %]" size="40" value="[% HTML.escape(language.translation) %]"></td>
   </tr>
   [%- END %]
   <tr>
    <th class="listheading"> </th>
    <th class="listheading">[% 'Remittance information optional Vendor/Customer No postfix' | $T8 %]</th>
   </tr>

   [%- FOREACH language = LANGUAGES %]
   <tr>
    <td>
     [%- IF language.id == 'default' %]
     [% 'Default (no language selected)' | $T8 %]
     [%- ELSE %]
     [%- HTML.escape(language.description) %]
     [%- END %]
    </td>
    <td><input name="translation__[% language.id %]__vc" size="40" value="[% HTML.escape(language.translation_vc) %]"></td>
   </tr>
   [%- END %]

  </table>
 </form>
```

A

```
[%- USE T8 %]
[%- USE HTML %]
<h1>[% HTML.escape(title) %]</h1>
[%- IF message %]
 <p>
 [% HTML.escape(message) %]
 </p>
[%- END %]
<form method="post" action="generictranslations.pl" id="form">
<table>
[%- FOREACH mail_string IN MAIL_STRINGS.keys.sort %]
  <tr>
    <th class="listheading"> </th>
    <th class="listheading">[% MAIL_STRINGS.$mail_string %]</th>
  </tr>

  [%- FOREACH language = LANGUAGES %]
   <tr>
    <td>
     [%- IF language.id == 'default' %]
      [% 'Default (no language selected)' | $T8 %]
     [%- ELSE %]
      [%- HTML.escape(language.description) %]
     [%- END %]
     </td>
     <td>
     [%- IF mail_string.search('preset') %]
      <textarea
        name="translation__[% language.id %]__[% mail_string %]" rows="4" cols="60">[% HTML.escape(language.$mail_string) %]
      </textarea>
     [%- ELSE %]
      <input
        name="translation__[% language.id %]__[% mail_string %]" size="40" value="[% HTML.escape(language.$mail_string) %]">
     [%- END %]
    </td>
   </tr>
  [%- END %]
[%- END %]
  </table>
 </form>
```

B

Oberfläche A

Begriffe bei SEPA-Überweisungen bearbeiten

	Verwendungszweckvorbelegung (Präfix)
Standard (keine Sprache ausgewählt)	Verwendungszweck
meine sprache	purpose
	optionale Verwendungszweckvorbelegung Kd./Lief.-Nummer
Standard (keine Sprache ausgewählt)	Lieferantenname
meine sprache	Supplier name

Vorbelegte Texte für E-Mails editieren

	Vorbelegter E-Mail-Text für Rechnungen
Standard (keine Sprache ausgewählt)	
meine sprache	
	Vorbelegter E-Mail-Text für Einkaufsaufträge
Standard (keine Sprache ausgewählt)	Einkaufsauftrag
meine sprache	hulibuli
	Vorbelegter E-Mail-Text für Anfragen
Standard (keine Sprache ausgewählt)	
meine sprache	
	Vorbelegter E-Mail-Text für Verkaufs-Lieferscheine
Standard (keine Sprache ausgewählt)	
meine sprache	

Oberfläche

B

	Vorbelegter E-Mail-Text für Aufträge
Standard (keine Sprache ausgewählt)	
meine sprache	
	Vorbelegter E-Mail-Text für Angebote
Standard (keine Sprache ausgewählt)	mein Angebot keine Ahnung wie das ansonsten aussieht usw. usf. Bla blub3 43
meine sprache	asdf33
	Anrede weiblich
Standard (keine Sprache ausgewählt)	
meine sprache	Mrs
	Anrede anonym (personenlos)
Standard (keine Sprache ausgewählt)	Sehr geehrte Damen und Herren
meine sprache	asdfasdfas
	Anrede männlich
Standard (keine Sprache ausgewählt)	
meine sprache	
	Zeichensetzungs-Trenner nach der Anrede-Formel (Punkt, Ausrufezeichen, etc)
Standard (keine Sprache ausgewählt)	,
meine sprache	:

Oberfläche

B

```
[%- USE T8 %]
[%- USE HTML %]
<h1>[% HTML.escape(title) %]</h1>

 [%- IF message %]
 <p>
  [% HTML.escape(message) %]
 </p>
 [%- END %]

 <form method="post" action="generictranslations.pl" id="form">

  <table>

   <tr>
    <th class="listheading"> </th>
    <th class="listheading">[% 'Remittance information prefix' | $T8 %]</th>
   </tr>

   [%- FOREACH language = LANGUAGES %]
   <tr>
    <td>
     [%- IF language.id == 'default' %]
     [% 'Default (no language selected)' | $T8 %]
     [%- ELSE %]
     [%- HTML.escape(language.description) %]
     [%- END %]
    </td>
    <td><input name="translation__[% language.id %]" size="40" value="[% HTML.escape(language.translation) %]"></td>
   </tr>
   [%- END %]

   <tr>
    <th class="listheading"> </th>
    <th class="listheading">[% 'Remittance information optional Vendor/Customer No postfix' | $T8 %]</th>
   </tr>

   [%- FOREACH language = LANGUAGES %]
   <tr>
    <td>
     [%- IF language.id == 'default' %]
     [% 'Default (no language selected)' | $T8 %]
     [%- ELSE %]
     [%- HTML.escape(language.description) %]
     [%- END %]
    </td>
    <td><input name="translation__[% language.id %]__vc" size="40" value="[% HTML.escape(language.translation_vc) %]"></td>
   </tr>
   [%- END %]

  </table>
```

A

```
[%- USE T8 %]
[%- USE HTML %]
<h1>[% HTML.escape(title) %]</h1>
[%- IF message %]
 <p>
 [% HTML.escape(message) %]
 </p>
[%- END %]
 <form method="post" action="generictranslations.pl" id="form">
  <table>
   [%- FOREACH mail_string IN MAIL_STRINGS.keys.sort %]
    <tr>
     <th class="listheading"> </th>
     <th class="listheading">[% MAIL_STRINGS.$mail_string %]</th>
    </tr>

    [%- FOREACH language = LANGUAGES %]
    <tr>
     <td>
      [%- IF language.id == 'default' %]
      [% 'Default (no language selected)' | $T8 %]
      [%- ELSE %]
      [%- HTML.escape(language.description) %]
      [%- END %]
     </td>
     <td>
      [%- IF mail_string.search('preset') %]
       <textarea name="translation__[% language.id %]__[% mail_string %]" rows="4" cols="60">[% HTML.escape(language.
$mail_string) %]</textarea>
      [%- ELSE %]
       <input name="translation__[% language.id %]__[% mail_string %]" size="40" value="[% HTML.escape(language.
$mail_string) %]">
      [%- END %]
     </td>
    </tr>
    [%- END %]
   [%- END %]
  </table>
 </form>
```

B

dynamische Attribute/Variablen

```
[%- FOREACH mail_string IN MAIL_STRINGS.keys.sort %]

  <input
     name="translation__[% language.id %]__[% mail_string %]"

     value="[% HTML.escape(language.$mail_string) %]"
  >

[%- END %]
```

```
sub edit_sepa_strings {
  $main::lxdebug->enter_sub();

  $main::auth->assert('config');

  my $form     = $main::form;
  my $locale   = $main::locale;

  $form->get_lists('languages' => 'LANGUAGES');

  my $translation_list = GenericTranslations->list(translation_type => 'sepa_remittance_info_pfx');
  my %translations     = map { ( ($_->{language_id} || 'default') => $_->{translation} ) } @{ $translation_list };

  my $translation_list_vc = GenericTranslations->list(translation_type => 'sepa_remittance_vc_no_pfx');
  my %translations_vc     =  map { ( ($_->{language_id} || 'default') => $_->{translation} ) } @{ $translation_list_vc };

  unshift @{ $form->{LANGUAGES} }, { 'id' => 'default', };

  foreach my $language (@{ $form->{LANGUAGES} }) {
    $language->{translation}    = $translations{$language->{id}};
    $language->{translation_vc} = $translations_vc{$language->{id}};
  }

  setup_generictranslations_edit_sepa_strings_action_bar();

  $form->{title} = $locale->text('Edit SEPA strings');
  $form->header();
  print $form->parse_html_template('generictranslations/edit_sepa_strings');

  $main::lxdebug->leave_sub();
}

sub save_sepa_strings {
  $main::lxdebug->enter_sub();

  $main::auth->assert('config');

  my $form     = $main::form;
  my $locale   = $main::locale;

  $form->get_lists('languages' => 'LANGUAGES');

  unshift @{ $form->{LANGUAGES} }, { };

  foreach my $language (@{ $form->{LANGUAGES} }) {
    GenericTranslations->save('translation_type' => 'sepa_remittance_info_pfx',
                              'translation_id'   => undef,
                              'language_id'      => $language->{id},
                              'translation'      => $form->{"translation__" . ($language->{id} || 'default')},);
    GenericTranslations->save('translation_type' => 'sepa_remittance_vc_no_pfx',
                              'translation_id'   => undef,
                              'language_id'      => $language->{id},
                              'translation'      => $form->{"translation__" . ($language->{id} || 'default') . "__vc" },);
  }

  $form->{message} = $locale->text('The SEPA strings have been saved.');

  edit_sepa_strings();

  $main::lxdebug->leave_sub();
}
```

```
sub edit_email_strings {
  $main::lxdebug->enter_sub();

  $main::auth->assert('config');

  my $form     = $main::form;
  my $locale   = $main::locale;

  $form->get_lists('languages' => 'LANGUAGES');
  unshift @{ $form->{LANGUAGES} }, { 'id' => 'default', };

  my (%translations, $translation_list);
  foreach (keys %mail_strings)  {
    $translation_list = GenericTranslations->list(translation_type => $_);
    %translations     = map { ( ($_->{language_id} || 'default') => $_->{translation} ) } @{ $translation_list };

    foreach my $language (@{ $form->{LANGUAGES} }) {
      $language->{$_} = $translations{$language->{id}};
    }
  }
  setup_generictranslations_edit_email_strings_action_bar();

  $form->{title} = $locale->text('Edit preset email strings');
  $form->header();
  print $form->parse_html_template('generictranslations/edit_email_strings',{ 'MAIL_STRINGS' => \%mail_strings });

  $main::lxdebug->leave_sub();
}

sub save_email_strings {
  $main::lxdebug->enter_sub();

  $main::auth->assert('config');

  my $form     = $main::form;
  my $locale   = $main::locale;

  $form->get_lists('languages' => 'LANGUAGES');

  unshift @{ $form->{LANGUAGES} }, { };
  foreach my $language (@{ $form->{LANGUAGES} }) {
    foreach (keys %mail_strings)  {
      GC->save('translation_type' => $_,
               'translation_id'   => undef,
               'language_id'      => $language->{id},
               'translation'      => $form->{"translation__" . ($language->{id} || 'default') . "__" . $_},
               );
    }
  }
  $form->{message} = $locale->text('The Mail strings have been saved.');

  edit_email_strings();

  $main::lxdebug->leave_sub();
}
```

B

```
sub edit_sepa_strings {
  $main::lxdebug->enter_sub();

  $main::auth->assert('config');

  my $form      = $main::form;
  my $locale    = $main::locale;

  $form->get_lists('languages' => 'LANGUAGES');

  my $translation_list = GenericTranslations->list(translation_type => 'sepa_remittance_info_pfx');
  my %translations     = map { ( ($_->{language_id} || 'default') => $_->{translation} ) } @{ $translation_list };

  my $translation_list_vc = GenericTranslations->list(translation_type => 'sepa_remittance_vc_no_pfx');
  my %translations_vc     =  map { ( ($_->{language_id} || 'default') => $_->{translation} ) } @{ $translation_list_vc };

  unshift @{ $form->{LANGUAGES} }, { 'id' => 'default', };

  foreach my $language (@{ $form->{LANGUAGES} }) {
    $language->{translation}    = $translations{$language->{id}};
    $language->{translation_vc} = $translations_vc{$language->{id}};
  }

  setup_generictranslations_edit_sepa_strings_action_bar();

  $form->{title} = $locale->text('Edit SEPA strings');
  $form->header();
  print $form->parse_html_template('generictranslations/edit_sepa_strings');

  $main::lxdebug->leave_sub();
}

sub save_sepa_strings {
  $main::lxdebug->enter_sub();

  $main::auth->assert('config');

  my $form      = $main::form;
  my $locale    = $main::locale;

  $form->get_lists('languages' => 'LANGUAGES');

  unshift @{ $form->{LANGUAGES} }, { };

  foreach my $language (@{ $form->{LANGUAGES} }) {
    GenericTranslations->save('translation_type' => 'sepa_remittance_info_pfx',
                              'translation_id'   => undef,
                              'language_id'      => $language->{id},
                              'translation'      => $form->{"translation__" . ($language->{id} || 'default')} );
    GenericTranslations->save('translation_type' => 'sepa_remittance_vc_no_pfx',
                              'translation_id'   => undef,
                              'language_id'      => $language->{id},
                              'translation'      => $form->{"translation__" . ($language->{id} || 'default') . "__vc" },);
  }

  $form->{message} = $locale->text('The SEPA strings have been saved.');

  edit_sepa_strings();

  $main::lxdebug->leave_sub();
}
```

A

```
sub edit_email_strings {
  $main::lxdebug->enter_sub();

  $main::auth->assert('config');

  my $form     = $main::form;
  my $locale   = $main::locale;

  $form->get_lists('languages' => 'LANGUAGES');
  unshift @{ $form->{LANGUAGES} }, { 'id' => 'default' };

  my (%translations, $translation_list);
  foreach (keys %mail_strings)  {
    $translation_list = GenericTranslations->list(translation_type => $_);
    %translations     = map { ( ($_->{language_id} || 'default') => $_->{translation} ) } @{ $translation_list };

    foreach my $language (@{ $form->{LANGUAGES} }) {
      $language->{$_} = $translations{$language->{id}};
    }
  }
  setup_generictranslations_edit_email_strings_action_bar();

  $form->{title} = $locale->text('Edit preset email strings');
  $form->header();
  print $form->parse_html_template('generictranslations/edit_email_strings',{ 'MAIL_STRINGS' => \%mail_strings });

  $main::lxdebug->leave_sub();
}

sub save_email_strings {
  $main::lxdebug->enter_sub();

  $main::auth->assert('config');

  my $form     = $main::form;
  my $locale   = $main::locale;

  $form->get_lists('languages' => 'LANGUAGES');

  unshift @{ $form->{LANGUAGES} }, { };
  foreach my $language (@{ $form->{LANGUAGES} }) {
    foreach (keys %mail_strings)  {
      GC->save('translation_type' => $_,
               'translation_id'   => undef,
               'language_id'      => $language->{id},
               'translation'      => $form->{"translation__" . ($language->{id} || 'default') . "__" . $_},
               );
    }
  }
  $form->{message} = $locale->text('The Mail strings have been saved.');

  edit_email_strings();

  $main::lxdebug->leave_sub();
}
```

B

SSRR (simpelSchleifenRedundanzReduzierer)

```
foreach (keys %mail_strings)  {
  $translation_list = GenericTranslations->list(translation_type => $_);
  %translations     = map { ( ($_->{language_id} || 'default')   =>
                              $_->{translation} ) }@{ $translation_list };

  foreach my $language (@{ $form->{LANGUAGES} }) {
    $language->{$_} = $translations{$language->{id}};
  }
}
```

```
foreach my $language (@{ $form->{LANGUAGES} }) {
  foreach (keys %mail_strings)  {
    GC->save('translation_type' => $_,
             'translation_id'   => undef,
             'language_id'      => $language->{id},
             'translation'      => $form->{"translation__" .
                                   ($language->{id} || 'default') .
                                    "__" . $_},
             );
  }
}
```

SSRR (simpelSchleifenRedundanzReduzierer)

```
foreach (keys %mail_strings)  {
  $translation_list = GenericTranslations->list(translation_type => $_);
  %translations     = map { ( ($_->{language_id} || 'default')   =>
                              $_->{translation} ) }@{ $translation_list };

  foreach my $language (@{ $form->{LANGUAGES} }) {
    $language->{$_} = $translations{$language->{id}};
  }
}
```

```
foreach my $language (@{ $form->{LANGUAGES} }) {
  foreach (keys %mail_strings)  {
    GC->save('translation_type' => $_,
             'translation_id'   => undef,
             'language_id'      => $language->{id},
             'translation'      => $form->{"translation__" .
                                   ($language->{id} || 'default') .
                                    "__" . $_},
             );
  }
}
```

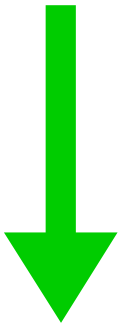
Kernidee SSRR

```
foreach (keys %mail_strings) {
  $translation_list = (..) $_;
}
```

```
my %mail_strings = (
  salutation_male    => t8('Salutation male'),
  salutation_female  => t8('Salutation female'),
  salutation_general => t8('Salutation general'),
);
```

Einfach erweiterbarer Code

```
my %mail_strings = (
  salutation_male    => t8('Salutation male'),
  salutation_female  => t8('Salutation female'),
  salutation_general => t8('Salutation general'),
);
```



```
my %mail_strings = (
  salutation_male     => t8('Salutation male'),
  salutation_female   => t8('Salutation female'),
  salutation_general  => t8('Salutation general'),
  preset_text_dunning => t8('Preset for dunnings'),
  preset_text_letter  => t8('Preset for letters'),
);
```

```
if ($form->{type} eq "sales_quotation") {
    ($path, $number) = ("angebote", $form->{quonumber});
  } elsif ($form->{type} eq "sales_order") {
    ($path, $number) = ("bestellungen", $form->{ordnumber});
  } elsif ($form->{type} eq "request_quotation") {
    ($path, $number) = ("anfragen", $form->{quonumber});
  } elsif ($form->{type} eq "purchase_order") {
    ($path, $number) = ("lieferantenbestellungen", $form->{ordnumber})
  } elsif ($form->{type} eq "sales_delivery_order") {
    ($path, $number) = ("verkaufslieferscheine", $form->{donumber});
  } elsif ($form->{type} eq "purchase_delivery_order") {
    ($path, $number) = ("einkaufslieferscheine", $form->{donumber});
  } elsif ($form->{type} eq "credit_note") {
    ($path, $number) = ("gutschriften", $form->{invnumber});
  } elsif ($form->{type} eq "letter") {
    ($path, $number) = ("briefe", $form->{letternumber});
  } elsif ($form->{vc} eq "customer") {
    ($path, $number) = ("rechnungen", $form->{invnumber});
  } elsif ($form->{vc} eq "vendor") {
    ($path, $number) = ("einkaufsrechnungen", $form->{invnumber});
  } else {
    $main::lxdebug->leave_sub();
    return undef;
  }
```

1. Typ => Pfadnamen

```
my %type_to_path = (
  sales_quotation          => 'angebote',
  sales_order              => 'bestellungen',
  request_quotation        => 'anfragen',
  purchase_order           => 'lieferantenbestellungen',
  sales_delivery_order     => 'verkaufslieferscheine',
  purchase_delivery_order  => 'einkaufslieferscheine',
  credit_note              => 'gutschriften',
  invoice                  => 'rechnungen',
  purchase_invoice         => 'einkaufsrechnungen',
  part                     => 'waren',
  service                  => 'dienstleistungen',
  assembly                 => 'erzeugnisse',
  letter                   => 'briefe',
  general_ledger           => 'dialogbuchungen',
  accounts_payable         => 'kreditorenbuchungen',
);
```

Hintergrund: http://hop.perl.plover.com/book/pdf/02DispatchTables.pdf

```
if ($form->{type} eq "sales_quotation") {
    ($path, $number) = ("angebote", $form->{quonumber});
  } elsif ($form->{type} eq "sales_order") {
    ($path, $number) = ("bestellungen", $form->{ordnumber});
  } elsif ($form->{type} eq "request_quotation") {
    ($path, $number) = ("anfragen", $form->{quonumber});
  } elsif ($form->{type} eq "purchase_order") {
    ($path, $number) = ("lieferantenbestellungen", $form->{ordnumber})
  } elsif ($form->{type} eq "sales_delivery_order") {
    ($path, $number) = ("verkaufslieferscheine", $form->{donumber});
  } elsif ($form->{type} eq "purchase_delivery_order") {
    ($path, $number) = ("einkaufslieferscheine", $form->{donumber});
  } elsif ($form->{type} eq "credit_note") {
    ($path, $number) = ("gutschriften", $form->{invnumber});
  } elsif ($form->{type} eq "letter") {
    ($path, $number) = ("briefe", $form->{letternumber});
  } elsif ($form->{vc} eq "customer") {
    ($path, $number) = ("rechnungen", $form->{invnumber});
  } elsif ($form->{vc} eq "vendor") {
    ($path, $number) = ("einkaufsrechnungen", $form->{invnumber});
  } else {
    $main::lxdebug->leave_sub();
    return undef;
  }
```

Sehr klare if-else Anweisung

```
# this is a good "if" use-case
int Min(int a, int b)
{
    if (a < b)
        return a;
    else
        return b;
}

# or, if you prefer the ternary operator
int Min(int a, int b)
{
    return (a < b) ? a : b;
}
```

Quelle: https://stackoverflow.com/questions/1554180/why-is-the-if-statement-considered-evil

Sehr unklare if else Verwendung

```
# this is called branching on a "type code",
# and screams for refactoring
void RunVehicle(Vehicle vehicle)
{
    # how the hell do I even test this?
    if (vehicle.Type == CAR)
        Drive(vehicle);
    elsif (vehicle.Type == PLANE)
        Fly(vehicle);
    else
        Sail(vehicle);
}
```

Quelle: https://stackoverflow.com/questions/1554180/why-is-the-if-statement-considered-evil

Ich erlaube dem Aufrufer zuviel!

-> Egal, was du mir gibst, ich krieg das zum Laufen

… bzw. wenn ich nicht genau weiß, was es ist, segel ich damit.

„The reason for this is that by allowing your callers to branch on a certain type code, you are creating a possibility to end up with numerous checks scattered all over your code, making extensions and maintenance much more complex. „

Quelle: https://stackoverflow.com/questions/1554180/why-is-the-if-statement-considered-evil

```
if ($form->{type} eq "sales_quotation") {
    ($path, $number) = ("angebote", $form->{quonumber});
  } elsif ($form->{type} eq "sales_order") {
    ($path, $number) = ("bestellungen", $form->{ordnumber});
  } elsif ($form->{type} eq "request_quotation") {
    ($path, $number) = ("anfragen", $form->{quonumber});
  } elsif ($form->{type} eq "purchase_order") {
    ($path, $number) = ("lieferantenbestellungen", $form->{ordnumber})
  } elsif ($form->{type} eq "sales_delivery_order") {
    ($path, $number) = ("verkaufslieferscheine", $form->{donumber});
  } elsif ($form->{type} eq "purchase_delivery_order") {
    ($path, $number) = ("einkaufslieferscheine", $form->{donumber});
  } elsif ($form->{type} eq "credit_note") {
    ($path, $number) = ("gutschriften", $form->{invnumber});
  } elsif ($form->{type} eq "letter") {
    ($path, $number) = ("briefe", $form->{letternumber});
  } elsif ($form->{vc} eq "customer") {
    ($path, $number) = ("rechnungen", $form->{invnumber});
  } elsif ($form->{vc} eq "vendor") {
    ($path, $number) = ("einkaufsrechnungen", $form->{invnumber});
  } else {
    $main::lxdebug->leave_sub();
    return undef;
  }
```

```
if ($form->{type} eq "sales_quotation") {
    ($path, $number) = ("angebote", $form->{quonumber});
  } elsif ($form->{type} eq "sales_order") {
    ($path, $number) = ("bestellungen", $form->{ordnumber});
  } elsif ($form->{type} eq "request_quotation") {
    ($path, $number) = ("anfragen", $form->{quonumber});
  } elsif ($form->{type} eq "purchase_order") {
    ($path, $number) = ("lieferantenbestellungen", $form->{ordnumber})
  } elsif ($form->{type} eq "sales_delivery_order") {
    ($path, $number) = ("verkaufslieferscheine", $form->{donumber});
  } elsif ($form->{type} eq "purchase_delivery_order") {
    ($path, $number) = ("einkaufslieferscheine", $form->{donumber});
  } elsif ($form->{type} eq "credit_note") {
    ($path, $number) = ("gutschriften", $form->{invnumber});
  } elsif ($form->{type} eq "letter") {
    ($path, $number) = ("briefe", $form->{letternumber});
  } elsif ($form->{vc} eq "customer") {
    ($path, $number) = ("rechnungen", $form->{invnumber});
  } elsif ($form->{vc} eq "vendor") {
    ($path, $number) = ("einkaufsrechnungen", $form->{invnumber});
  } else {
    $main::lxdebug->leave_sub();
    return undef;
  }
```

```
if ($form->{type} eq "sales_quotation") {
    ($path, $number) = ("angebote", $form->{quonumber});
  } elsif ($form->{type} eq "sales_order") {
    ($path, $number) = ("bestellungen", $form->{ordnumber});
  } elsif ($form->{type} eq "request_quotation") {
    ($path, $number) = ("anfragen", $form->{quonumber});
  } elsif ($form->{type} eq "purchase_order") {
    ($path, $number) = ("lieferantenbestellungen", $form->{ordnumber})
  } elsif ($form->{type} eq "sales_delivery_order") {
    ($path, $number) = ("verkaufslieferscheine", $form->{donumber});
  } elsif ($form->{type} eq "purchase_delivery_order") {
    ($path, $number) = ("einkaufslieferscheine", $form->{donumber});
  } elsif ($form->{type} eq "credit_note") {
    ($path, $number) = ("gutschriften", $form->{invnumber});
  } elsif ($form->{type} eq "letter") {
    ($path, $number) = ("briefe", $form->{letternumber});
  } elsif ($form->{vc} eq "customer") {
    ($path, $number) = ("rechnungen", $form->{invnumber});
  } elsif ($form->{vc} eq "vendor") {
    ($path, $number) = ("einkaufsrechnungen", $form->{invnumber});
  } else {
    $main::lxdebug->leave_sub();
    return undef;
  }
```

2. Dein Problem bleibt dein Problem

```
sub webdav_folder {
  my ($self) = @_;

  croak "No client set in \$::auth" unless $::auth->client;
  croak "Need number"               unless $self->number;

  my $type = $type_to_path{$self->type};

  croak "Unknown type"              unless $type;

  # ergebnis: einkaufsrechnungen mit Nummer 39 und Mandantennummer 3
```

Hintergrund: http://www.perlmonks.org/?node_id=685452

3. Fehlerfall abgefangen

```
sub webdav_folder {
  my ($self) = @_;


  croak "No client set in \$::auth" unless $::auth->client;
  croak "Need number"               unless $self->number;



  my $type = $type_to_path{$self->type};

  croak "Unknown type"              unless $type;

  # ergebnis: einkaufsrechnungen mit Nummer 39 und Mandantennummer 3
```

Hintergrund: http://www.perlmonks.org/?node_id=685452

Kernidee

```
# einmalig vorkommenden datenstruktur!
# do not copy & waste this one!!
my %type_to_path = (
   sales_quotation          => 'angebote',
   sales_order              => 'bestellungen',
   request_quotation        => 'anfragen',
);
```

```
   my $type = $type_to_path{$self->type};
```

Svens Signatur: git grep "my %type_to_path"

Beispiel

```
my $type = $type_to_path{'sales_order'};
echo $type; # bestellungen
```

Code in Value

```
$mod_to_no = { ap => sub { $record_type = 'accounts_payable';
                           $number      = $ref>{reference}    },
               gl => sub { $record_type = 'general_ledger';
                           $number = $ref->{trans_id}         },
               ar => sub { $record_type = 'purchase_invoice';
                           $number = $ref->{reference}        },
               is => sub { $record_type = 'invoice';
                           $number = $ref->{reference}        },
             };

# if we have a module that fits a record type
if (exists $mod_to_no->{$ref->{module}}) {
  $mod_to_no->{$ref->{module}}->();
}
```

Hintergrund: http://hop.perl.plover.com/book/pdf/02DispatchTables.pdf

Code in Value

```
$mod_to_no = { ap => sub { $record_type = 'accounts_payable';
                           $number      = $ref>{reference}    },
               gl => sub { $record_type = 'general_ledger';
                           $number = $ref->{trans_id}         },
               ar => sub { $record_type = 'purchase_invoice';
                           $number = $ref->{reference}        },
               is => sub { $record_type = 'invoice';
                           $number = $ref->{reference}        },
             };

# if we have a module that fits a record type
if (exists $module_to_number->{$ref->{module}}) {
  $module_to_number->{$ref->module}}->();
}
```

Code ist klar erweiterbar

```
$mod_to_no = { ap => sub { $record_type = 'accounts_payable';
                           $number      = $ref>{reference}   },
               gl => sub { $record_type = 'general_ledger';
                           $number = $ref->{trans_id}        },
               ar => sub { $record_type = 'purchase_invoice';
                           $number = $ref->{reference}       },
               is => sub { $record_type = 'invoice';
                           $number = $ref->{reference}       },
               oe => sub { $record_type = 'order';
                           $number = $ref->{reference}       },
             };

# if we have a module that fits a record type
if (exists $mod_to_no->{$ref->{module}}) {
  $mod_to_no->{$ref->{module}}->();
}
```

Code ist wiederaufrufbar!

```
$mod_to_no = { ap => sub { $record_type = 'accounts_payable';
                           $number      = $ref>{reference}   },
               gl => sub { $record_type = 'general_ledger';
                           $number = $ref->{trans_id}        },
               ar => sub { $record_type = 'purchase_invoice';
                           $number = $ref->{reference}       },
               is => sub { $record_type = 'invoice';
                           $number = $ref->{reference}       },
               oe => sub { $record_type = 'order';
                           $number = $ref->{reference}       },
               ag => sub { $mod_to_no->{ap}->();
                           print "$record_type erfolgreich!";
                           $mod_to_no->{gl}->();             },
             };

# if we have a module that fits a record type
if (exists $mod_to_no->{$ref->{module}}) {
  $mod_to_no->{$ref->{module}}->();
}
```

Code ist kombinierbar!

```
$mod_to_no = { ap => sub { $record_type = 'accounts_payable';
                           $number      = $ref>{reference}    },
               gl => sub { $record_type = 'general_ledger';
                           $number = $ref->{trans_id}         },
               ar => sub { $record_type = 'purchase_invoice';
                           $number = $ref->{reference}        },
               is => sub { $record_type = 'invoice';
                           $number = $ref->{reference}        },
               oe => sub { $record_type = 'order';
                           $number = $ref->{reference}        },
               ag => sub { $mod_to_no->{ap}->();
                           echo "$record_type erfolgreich!";
                           $mod_to_no->{gl}->();              },
             };

# if we have a module that fits a record type
if (exists $mod_to_no->{$ref->{module}}) {
  $mod_to_no->{$ref->{module}}->();
}
```

rsync host1 host2

- Zustand host1 unbekannt
- Zustand host2 unbekannt
- Status Netzkommunikation unbekannt
- Manuelle Kontrolle erforderlich

Rsync ist wiedereintrittssicher!

- Dateien werden inkrementell kopiert
- Diverse Parameter (--size-only)

=> Nur aktuelle Differenz wird kopiert

Rsync Aufruf aus Perl

```
my ($out, $err) = capture {
                    system("/usr/bin/rsync --rsync-path='sudo rsync
                          --verbose --times --owner --group -ignore-times
                          --links  --perms   -r --size-only --delete -force
                          --numeric-ids   --stats -e 'ssh -c aes128-ctr'
                          test\@meinserver.de:$params{src}
                          $params{dest}");
                           };
```

Rekursiver Aufruf innerhalb Funktion

```
sub rsync_recursive {
  my %params = @_;
  my $debug  = 0;

  croak "No good source dir for linux!"            unless ($params{src} =~ m/[^\0]+/);
  croak "Damn not a real destination dir here!"    unless (-d $params{dst});

  print "Synce von $params{src} nach $params{dst} \n" if $debug;

  my ($out, $err) = capture {
                      system("/usr/bin/rsync --rsync-path='sudo rsync'
                             --verbose --times --owner --group -ignore-times
                             --links  --perms   -r --size-only --delete -force
                             --numeric-ids   --stats -e 'ssh -c aes128-ctr'
                             test\@meinserver.de:$params{src} $params{dest}");
                           };

  # recursively try again
  if ($err =~ m/message authentication code incorrect/) {
    print "recursion reached with : $err" if $debug;
    sleep 3;
    rsync_recursive();
  }
  return $out;
}
```

Initialer Aufruf

```
my $return = rsync_recursive(src => "/var/www/", dst => "/var/www/");

# expect this good value
if ($return !~ m/total size is [0-9,]*  speedup is [0-9.]*/) {
  $success = "false";
  $reason  = "Rsync-Fehler mit Rückgabe: $return";
} else {
  print "alles i.O.!" if $debug;
}
```

Fragen ?
Quellen !
Kontakt.

- Pragmatisch Programmieren (Hunt)
- Higher Order Perl (Dominus)
- StackOverflow
- Github (Projekt kivitendo)

Mail: jan@kivitendo-premium.de