

Perl::Critic

```
#!/usr/bin/perl
```

```
$f = <STDIN>;
```

```
chomp $f;
```

```
$f += 4 if $f == 2;
```

```
for( $i = 0; $i <= $f; $i++ ){
```

```
    print($i);
```

```
}
```

C:\ Eingabeaufforderung

```
C:\Dokumente und Einstellungen\Renee\Eigene Dateien\Perl\community>perl_critic.p  
1  
Code before strictures are enabled at line 3, column 1. See page 429 of PBP.
```

C:\ Eingabeaufforderung

```
C:\Dokumente und Einstellungen\Renee\Eigene Dateien\Perl\community>perl_critic.p  
1  
RCS keywords $Id$ not found at line 1, column 1. See page 441 of PBP.  
RCS keywords $Revision$, $HeadURL$, $Date$ not found at line 1, column 1. See pa  
ge 441 of PBP.  
RCS keywords $Revision$, $Source$, $Date$ not found at line 1, column 1. See pag  
e 441 of PBP.  
No "VERSION" variable found at line 1, column 1. See page 404 of PBP.  
Code before strictures are enabled at line 3, column 1. See page 429 of PBP.  
Code before warnings are enabled at line 3, column 1. See page 431 of PBP.  
Postfix control "if" used at line 6, column 9. See pages 93,94 of PBP.  
C-style "for" loop used at line 8, column 5. See page 97 of PBP.
```

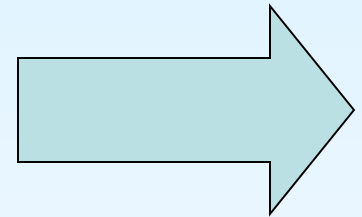
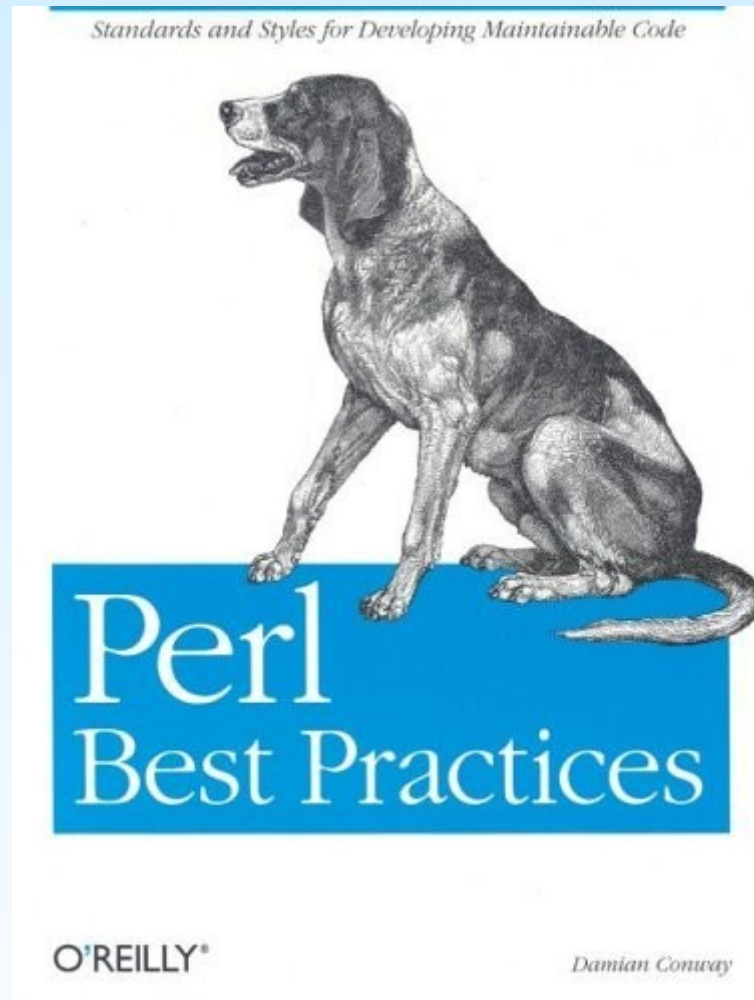
Was ist Perl::Critic?

- Modul zum Überprüfen des Programmierstils
- Setzt „Regeln“ aus Perl Best Practices um
 - !Kein! Programmier-„Gesetz“
 - Damian`s Ansicht von gutem Programmierstil
 - Teilt die Regeln in „Grade“
- Test::Perl::Critic
- Eigene Policies möglich
- Basiert auf PPI

Warum Perl::Critic?

- Einheitlicher Code
 - Wartbarer
 - Erleichtert Arbeiten in Teams
- Verbessern und Festigen des eigenen Programmierstils

„Regeln“



Perl::Critic::Policy::BuiltinFunctions::RequireBlockMap	1.080
Perl::Critic::Policy::BuiltinFunctions::RequireGlobFunction	1.080
Perl::Critic::Policy::BuiltinFunctions::RequireSimpleSortBlock	1.080
Perl::Critic::Policy::ClassHierarchies::ProhibitAutoloading	1.080
Perl::Critic::Policy::ClassHierarchies::ProhibitExplicitISA	1.080
Perl::Critic::Policy::ClassHierarchies::ProhibitOneArgBless	1.080
Perl::Critic::Policy::CodeLayout::ProhibitHardTabs	1.080
Perl::Critic::Policy::CodeLayout::ProhibitParensWithBuiltins	1.080
Perl::Critic::Policy::CodeLayout::ProhibitQuotedWordLists	1.080
Perl::Critic::Policy::CodeLayout::ProhibitTrailingWhitespace	1.080
Perl::Critic::Policy::CodeLayout::RequireConsistentNewlines	1.080
Perl::Critic::Policy::CodeLayout::RequireTidyCode	1.080
Perl::Critic::Policy::CodeLayout::RequireTrailingCommas	1.080
Perl::Critic::Policy::ControlStructures::ProhibitCStyleForLoops	1.080
Perl::Critic::Policy::ControlStructures::ProhibitCascadingIfElse	1.080
Perl::Critic::Policy::ControlStructures::ProhibitDeepNests	1.080
Perl::Critic::Policy::ControlStructures::ProhibitMutatingListFunctions	1.080
Perl::Critic::Policy::ControlStructures::ProhibitNegativeExpressionsInUnlessAndUntilConditions	1.080
Perl::Critic::Policy::ControlStructures::ProhibitPostfixControls	1.080
Perl::Critic::Policy::ControlStructures::ProhibitUnlessBlocks	1.080
Perl::Critic::Policy::ControlStructures::ProhibitUnreachableCode	1.080
Perl::Critic::Policy::ControlStructures::ProhibitUntilBlocks	1.080
Perl::Critic::Policy::Documentation::PodSpelling	1.080
Perl::Critic::Policy::Documentation::RequirePodAtEnd	1.080
Perl::Critic::Policy::Documentation::RequirePodSections	1.080
Perl::Critic::Policy::ErrorHandling::RequireCarping	1.080
Perl::Critic::Policy::InputOutput::ProhibitBacktickOperators	1.080
Perl::Critic::Policy::InputOutput::ProhibitBarewordFileHandles	1.080
Perl::Critic::Policy::InputOutput::ProhibitExplicitStdin	1.080
Perl::Critic::Policy::InputOutput::ProhibitInteractiveTest	1.080
Perl::Critic::Policy::InputOutput::ProhibitJoinedReadline	1.080

Eigene Richtlinien

- Firmen haben für Java,C,... eigene Programmierrichtlinien
- ... aber keine für Perl → führt zu einem Wildwuchs im Programm
- Damian`s Regeln manchmal gegen eigene Vorstellungen
- Beispiel
 - RequireSystemList
 - RequireCurlyBracketInLine

Exkurs: PPI

Perl parsen...

Was fällt uns dazu ein?

- Only perl can parse Perl
- Only perl can parse Perl
- Only perl can parse Perl
 - Only perl can parse Perl
 - Only perl can parse Perl
 - Only perl can parse Perl
 - » Only perl can parse Perl

perl can only parse Perl

- Stimmt das?

perl sucks

- Was ist Perl?

Das ist Perl (für perl)

```
use Acme::Buffy;
BUffY bUFFY BUffY bUFFY bUffY buFFY
bUffY bUFFY buffy buffy bUffY buFFY buffY
BuFfybUffY buFFY BuFFY buffy bUffY buFFY
buffY BuFFY buFfybUFFY bUffY buffY buFFY
BUffY BUffY BuFFY BUffY BUffY BUffY
BuFFYbUffY BUffY BUffY buFFYBUffY
BUffY BuFFY BUffY BUffY bUffY buFFY
buffy BUffY bUffY BUffY bu
```

Das ist Perl (für perl)

```
use Acme::DoubleHelix;
```

```
CG
T--A
A---T
A-----T
C-----G
T-----A
A---T
G--C
AT
CG
C--G
G---C
```

Das ist Perl (für alle)

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```

Was ist parsen

- Ist für jeden unterschiedlich
 - Zwei Menschen – zwei Meinungen
 - Maschine != Mensch

Was ist parsen

Ein **Parser** (**engl.** *to parse* „analysieren“ bzw. von **lateinisch** *pars* „Teil“; weshalb Parser im Deutschen gelegentlich auch als *Zerteiler* bezeichnet werden) ist ein **Computerprogramm**, das entscheidet, ob ein Eingabetext zur **formalen Sprache** einer bestimmten **Grammatik** gehört.

So parst perl

→ `#!/usr/bin/perl`

`use strict;`

`use warnings;`

`use PPI;`

`my $ppi = PPI::Document->new($0);`

`my $dumper = PPI::Dumper->new($ppi);`

`$dumper->print();`

Pfad des Interpreters

So parst perl

```
#!/usr/bin/perl
```



```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```

So parst perl

```
#!/usr/bin/perl
```

→ **use** strict;
use warnings;
use PPI;

Suche strict.pm und
führe strict.pm aus –
springt aus Programm
raus

```
my $ppi = PPI::Document->new($0);  
my $dumper = PPI::Dumper->new($ppi);  
$dumper->print();
```

So parst perl

```
#!/usr/bin/perl
```

```
use strict;  
→ use warnings;  
use PPI;
```

Suche warnings.pm und
führe warnings.pm aus –
springt aus Programm
raus

```
my $ppi = PPI::Document->new($0);  
my $dumper = PPI::Dumper->new($ppi);  
$dumper->print();
```

So parst perl

```
#!/usr/bin/perl
```

```
use strict;  
use warnings;  
→ use PPI;
```

Suche PPI.pm und führe
PPI.pm aus – springt aus
Programm raus

```
my $ppi = PPI::Document->new($0);  
my $dumper = PPI::Dumper->new($ppi);  
$dumper->print();
```

So parst perl

```
#!/usr/bin/perl
```

```
use strict;  
use warnings;  
use PPI;
```

```
→ my $ppi = PPI::Document->new($0);  
my $dumper = PPI::Dumper->new($ppi);  
$dumper->print();
```



Erzeuge neues
PPI::Document-Objekt

Menschliches parsen

- Nur auf ein Dokument bezogen
- Inhalt des Dokuments analysieren

Was ist PPI

- PPI = Perl::Parse::Isolated
- parst Perl-Dokumente nach menschlichem Verständnis
- baut einen Baum auf
 - Ermöglicht das Durchwandern für die Suche nach Knoten (Token)
 - Ausgabe über PPI::Dumper
- nützlich zur Quellcode-Analyse

So parst PPI

Kommentar

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```

So parst PPI

```
#!/usr/bin/perl
```

Whitespace (\n)

```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```

So parst PPI

```
#!/usr/bin/perl
```

Whitespace (\n)

```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```



Include

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```



Word

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```



Whitespace ()

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```



Word

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;  
use warnings;  
use PPI;
```



Structure (;)

```
my $ppi = PPI::Document->new($0);  
my $dumper = PPI::Dumper->new($ppi);  
$dumper->print();
```

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```



Variable

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```



Word

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```



Whitespace ()

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```



Symbol

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```



Whitespace ()

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```



Operator

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```



Word

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```



Operator

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

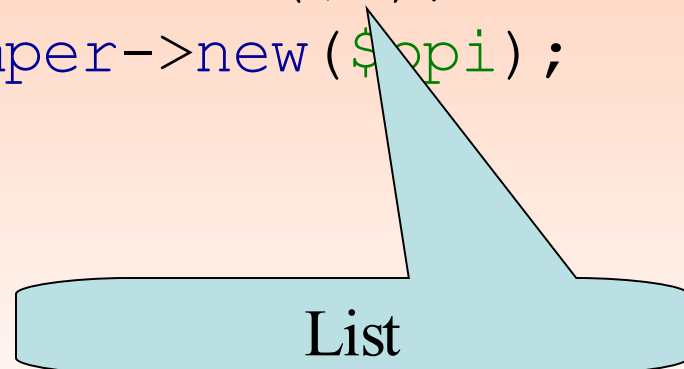
```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```



So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```



Magic

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```



Structure (;)

So parst PPI

```
#!/usr/bin/perl
```

```
use strict;
```

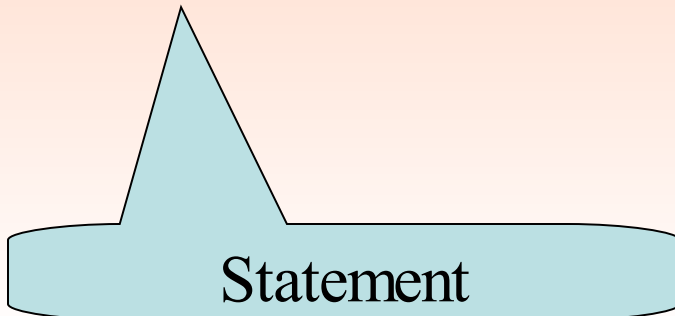
```
use warnings;
```

```
use PPI;
```

```
my $ppi = PPI::Document->new($0);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```



So parst PPI das Programm

```
PPI::Document
  PPI::Token::Comment    '#!/usr/bin/perl\n'
  PPI::Token::Whitespace '\n'
  PPI::Statement::Include
    PPI::Token::Word     'use'
    PPI::Token::Whitespace ' '
    PPI::Token::Word     'strict'
    PPI::Token::Structure ';'
  PPI::Token::Whitespace '\n'
[...]
PPI::Statement::Variable
  PPI::Token::Word      'my'
  PPI::Token::Whitespace ' '
  PPI::Token::Symbol    '$ppi'
  PPI::Token::Whitespace ' '
  PPI::Token::Operator  '='
  PPI::Token::Whitespace ' '
  PPI::Token::Word      'PPI::Document'
  PPI::Token::Operator  '->'
  PPI::Token::Word      'new'
  PPI::Structure::List  ( ... )
    PPI::Statement::Expression
      PPI::Token::Magic '$0, [...]
```

RequireCurlyBracketsInLine

- Öffnende geschweifte Klammer soll in der Zeile mit dem Schleifenkopf sein.

Analyse der eigenen Regel

- Alle Möglichkeiten aufschreiben:

```
for my $var ( @array ) {  
    # ...  
}
```



```
for ( @array ) {  
    # ...  
}
```



```
for my $var ( @array )  
{  
    # ...  
}
```



```
print $_ for ( @array );
```



PPI::Dumper

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```

```
use PPI;
```

```
my $file = './for_analyse.pl';
```

```
my $ppi = PPI::Document->new($file);
```

```
my $dumper = PPI::Dumper->new($ppi);
```

```
$dumper->print();
```

PPI::Dumper

```
for my $var ( @array ) {  
    # ...  
}
```

PPI::Statement::Compound

```
PPI::Token::Word      'for'  
PPI::Token::Whitespace ' '  
PPI::Token::Word      'my'  
PPI::Token::Whitespace ' '  
PPI::Token::Symbol     '$var'  
PPI::Token::Whitespace ' '  
PPI::Structure::ForLoop ( ... )  
    PPI::Statement  
        PPI::Token::Symbol '@array'  
PPI::Structure::Block  { ... }
```

PPI::Dumper

```
for ( @array ) {  
    # ...  
}
```

PPI::Statement::Compound

```
PPI::Token::Word      'for'  
PPI::Token::Whitespace ' '  
PPI::Structure::ForLoop ( ... )  
    PPI::Statement  
        PPI::Token::Symbol '@array'  
PPI::Structure::Block { ... }
```

PPI::Dumper

```
for ( @array )  
{  
    # ...  
}
```

PPI::Statement::Compound

```
PPI::Token::Word      'for'  
PPI::Token::Whitespace ' '  
PPI::Structure::ForLoop ( ... )  
    PPI::Statement  
        PPI::Token::Symbol '@array'  
PPI::Token::Whitespace '\n'  
PPI::Structure::Block { ... }
```

PPI::Dumper

```
print $_ for ( @array );
```

PPI::Statement

```
PPI::Token::Word      'print'  
PPI::Token::Whitespace ' '  
PPI::Token::Magic    '$_  
PPI::Token::Whitespace ' '  
PPI::Token::Word      'for'  
PPI::Token::Whitespace ' '  
PPI::Structure::ForLoop ( ... )  
    PPI::Token::Whitespace ' '  
    PPI::Statement  
        PPI::Token::Symbol '@array'  
        PPI::Token::Whitespace ' '  
PPI::Token::Structure ';'
```

Umsetzung der eigenen Regel

```
package Perl::Critic::Policy::ControlStructures::GPW;  
  
use strict;  
use warnings;  
use Perl::Critic::Utils;  
use Perl::Critic::Violation;  
  
use base qw(Perl::Critic::Policy);  
  
our $VERSION = '0.1';
```

Umsetzung der eigenen Regel

```
my $desc = q~Die öffnende geschweifte Klammer  
ist nicht in der Zeile des Schleifenkopfes.~;
```

```
my $expl = q~Die öffnende Klammer muss in der Zeile  
des Zeilenkopfes stehen, also
```

```
    for my $var ( ... ){  
        ...  
    }
```

anstatt

```
    for my $var ( ... )  
    {  
        ...  
    }  
~;
```

Umsetzung der eigenen Regel

```
sub applies_to {  
    return 'PPI::Statement::Compound'  
}
```

```
sub default_themes {  
    return qw/core foo/  
}
```

```
sub default_severity {  
    return $SEVERITY_HIGHEST  
}
```

PPI::Dumper

```
for my $var ( @array ) {  
    # ...  
}
```

PPI::Statement::Compound

```
PPI::Token::Word      'for'  
PPI::Token::Whitespace ' '  
PPI::Token::Word      'my'  
PPI::Token::Whitespace ' '  
PPI::Token::Symbol     '$var'  
PPI::Token::Whitespace ' '  
PPI::Structure::ForLoop ( ... )  
    PPI::Statement  
        PPI::Token::Symbol '@array'  
PPI::Structure::Block  { ... }
```

Umsetzung der eigenen Regel

```
sub applies_to {  
    return 'PPI::Statement::Compound'  
}
```

```
sub default_themes {  
    return qw/core foo/  
}
```

```
sub default_severity {  
    return $SEVERITY_HIGHEST  
}
```

Umsetzung der eigenen Regel

```
sub violates{  
  my ($self, $elem, $doc) = @_  
  
  # Analyse des Codes  
  # Verwendung von PPI  
  # Zwischen ForLoop und Block muss  
  # ein Newline kommen  
  
  if( $newline_between_forloop_and_block ){  
    my $sev = $self->get_severity;  
    return Perl::Critic::Violation->new(  
      $desc, $expl, $elem, $sev  
    );  
  }  
  return;
```

```
}
```

Umsetzung der eigenen Regel

```
my $base = $elem->schild(0);
return unless $base eq 'for';

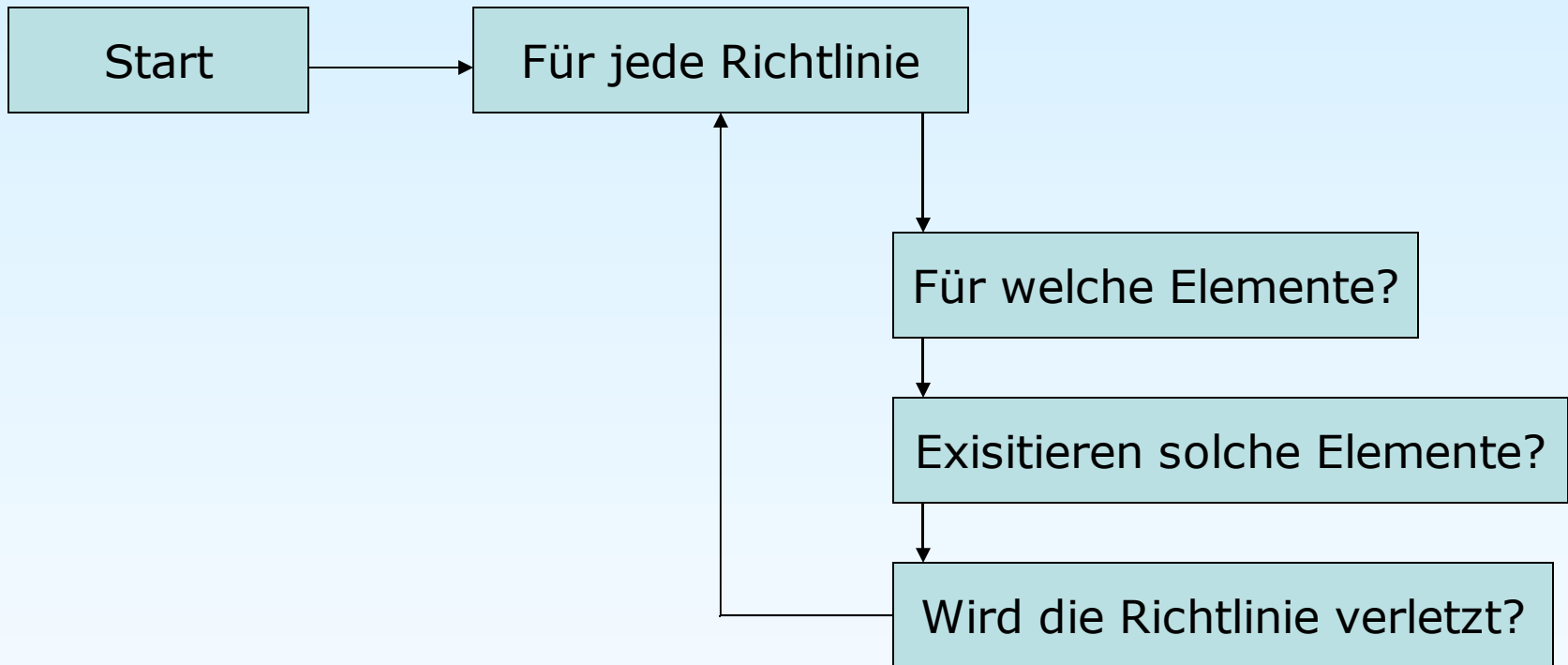
my ($list, $newline, $block);

my @children = $elem->children;

for my $i ( 0 .. $#children ) {
    my $schild = $children[$i];

    if( $schild->isa( 'PPI::Structure::Block' ) ) {
        $block = $i;
    }
    # ...
}
```

Im Hintergrund...



Test

```
#!/usr/bin/perl
```

```
use strict;
```

```
my $f = 3;
```

```
for( my $i = 0; $i <= $f; $i++ )
```

```
{
```

```
    print($i);
```

```
}
```

Test

```
C:\> Eingabeaufforderung
C:\Dokumente und Einstellungen\Renee\Eigene Dateien\Perl\community>perl_critic.p
1
Die ÷ffnende geschweifte Klammer
ist nicht in der Zeile des Schleifenkopfes. at line 6, column 1. Die ÷ffnende K
lammer muss in der Zeile
des Zeilenkopfes stehen, also

    for my $var < ... >{
        ...
    }

anstatt

    for my $var < ... >
    {
        ...
    }
.
```

Weiterführende Infos

- Perl::Critic ([http://search.cpan.org/dist/**Perl-Critic**/](http://search.cpan.org/dist/Perl-Critic/))
- <http://www.slideshare.net/joshua.mcadams/an-in>
- <http://www.slideshare.net/joshua.mcadams/yapci>

