

Migrating code with SmaCC

John Brant

brant@refactoryworkers.com

Migration Strategy

- Define Parser

SmaCC

- Create transformation program

- Compatibility layer

✓ Normal development continues

- Keeps same design — garbage in garbage out

Parser Definition

- LALR(1)/LR(1) parsers
- GLR
- AST generation
- Pattern matching

AST Definition

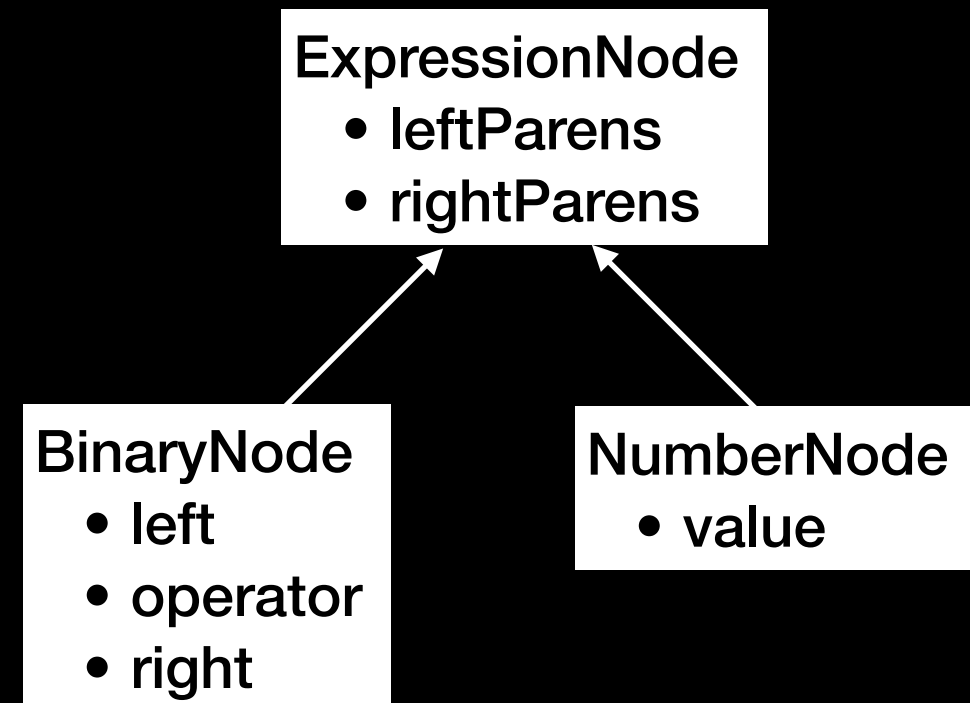
<number> : \d+ (\. \d*)? ;

<whitespace> : \s+;

%left "+";

%root Expression;

%suffix Node;



Expression

: Expression 'left' "+" 'operator' Expression 'right' {{Binary}}
| "(" 'leftParen' Expression ")" 'rightParen' {{Expression}}
| <number> 'value' {{Number}}
;

Transformation Program

- Ordered list of transformation rules + methods and properties
- Declarative Pattern Rules
 - Quick to write
 - One-off expressions
- Imperative Code Rules
 - General syntax
 - Control flow

Pattern Rules

- Search expression pattern-based AST
- Replace expression is pattern-based string

Pattern Matching

```
<number> : \d+ (\. \d*) ? ;  
<whitespace> : \s+;  
<patternToken> : `[^\`]+`;
```

```
%glr;  
%left "+";  
%root Expression;  
%suffix Node;
```

Expression

```
: Expression 'left' "+" 'operator' Expression 'right' {{Binary}}  
| "(" 'leftParen' Expression ")" 'rightParen' {{Expression}}  
| <number> 'value' {{Number}}  
;
```

``a` + `a`  $\Rightarrow$  `a` * 2`

Patterns can
match any AST
node

Pattern Example

Source:

3 + 3

BinaryNode: +

NumberNode:
3

NumberNode:
3

Search Pattern:

`a` + `a`

BinaryNode: +

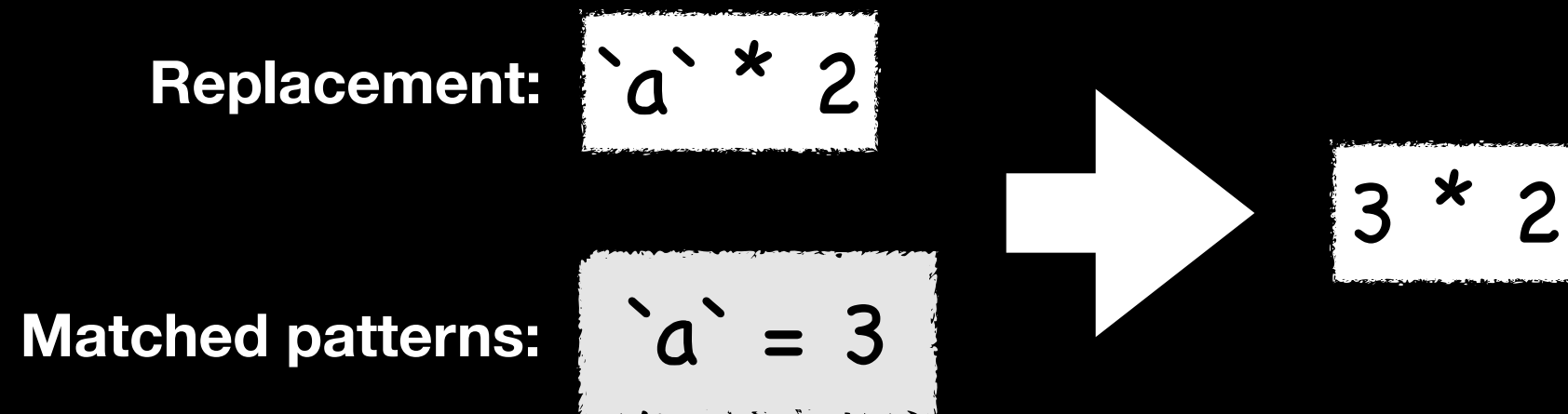
Anything: `a`

Anything: `a`

`a` = 3

Replace Expressions

- Replacement pattern is string macro
- Original source replaced with expanded macro
- Matched pattern nodes rewritten before replacement string is generated



Pattern Examples

for `a` := `b` to `c` - 1 do `d` $\Rightarrow$

for (`a` = `b`; `a` < `c`; `a`++) `d`

`a`/Forms.TCustomForm`.Constraints.MinHeight := `b` $\Rightarrow$

`a`.MinimumSize = new Size(`a`.MinimumSize.Width, `b`)

Code Rules

- Smalltalk expressions
- Search expressions based on AST node and code
- Replace expressions
 - ★ Edit expressions
 - ★ Control flow
 - ★ General Smalltalk code

Edit Expressions

- Custom framework messages for editing source
- Replacing
#replace:with: #replaceAll:with: ...
- Moving
#move:before: #move:after ...
- Inserting
#insert:before: #insert:afterAll: ...
- Deleting
#delete: #deleteWithWhitespaceAfter: ...

Control Flow

- Normal traversal is depth first
- Change the order that nodes are traversed
#processChild: #processChildren #continue

Code Examples

"{ }"

For: DelphiStatementBlockNode

When:

 true

Do:

 self replace: match beginToken with: '{'.

 self replace: match endToken with: '}'.

 self continue

"function objects"

For: PBTypeDeclarationNode

When:

 match from source sameAs: 'function_object'

Do:

 self isStatic: true.

 self classStart: match startPosition.

 self

 replace: match

 with: 'public partial class ' , self functionsClassName , ' {'

Parser Debugger

debug it

Bytecode GT

Parser Stack

Through Action Into Reduce Next Token Possible Actions

GLR

Symbol	Value
Lookahead:	
<identifier>;<identifierName>	{url(122,124,#(97 98))}
" , "	{,(119,119,#(27))}
VariableDeclarationList	an Array(an OrderedCollection(a JSVariableDeclarationNode) an Or
"var"	{var(93,95,#(91 97 98))}
ModuleItemList	an OrderedCollection(a JSVariableStatementNode a JSVariableStat

Symbol	Action
"["	Shift
"{"	Shift
<identifier>	Shift
BindingPattern	Shift
Identifier	Shift
VariableDeclaration	Shift
ObjectBindingPattern	Shift
ArrayBindingPattern	Shift

1

Input

Step to Cursor Scanner Step Next Character

```
var sqlite = require('sqlite3');
var db = new sqlite.Database('/var/lib/weewx/weewx.sdb');

var http = require('http'),
    url = require('url'),
    path = require('path'),
    fs = require('fs');
const PORT = 8080;
const staticDirectory = path.join(process.cwd(), 'files');

function convertToDate(value) {
    return new Date(value * 1000);
}

function convertToTs(date) {
```

Name	Value
Scope	default

Stack

Proceed Restart Into Over Through Source Where is? Browse

JSParser(SmaCCGLRParser)	getNextToken
JSParser(SmaCCGLRParser)	performParsingLoop
JSParser(SmaCCParser)	parse
JSParser class(SmaCCParser class)	parseStream:startingAt:
JSParser class(SmaCCParser class)	parseFile:encoding: [:stream stream
FileReference(AbstractFileReference)	readStreamDo: [aBlock value: stre
BlockClosure	ensure:

```
getNextToken
    currentToken isNil
        ifFalse: [ ^ self ].
    lastState = currentState scannerState
    ifTrue: [ currentToken := lastToken.
        scanner restoreState: nextScannerState.
        currentState scannerState: nextScannerState ]
```

Previewing

SmaCC Transformation Toolkit

Configuration Transformations Preview

w_eventhandling.srw w_eventhandling.cs w_eventhandling.Desi... w_eventhandling.resx

```
87 fontpitch fontpitch = variable!  
88 fontfamily fontfamily = swiss!  
89 string facename = "Tahoma"  
90 string text = "clear"  
91 end type  
92  
93 event clicked;  
94 clear()  
95 end event  
96  
97 type mle_console from multilineedit within w_eventhandling  
98 integer x = 242  
99 integer y = 1448  
100 integer width = 2235  
101 integer height = 1324  
102 integer taborder = 20  
103 integer textsize = -10  
104 integer weight = 400
```

```
31  
32 public virtual void Logevent () {  
33 }  
34  
35 public virtual void this_Open(object sender, EventArgs e) {  
36 dw_pr.SetTransObject(SpclTransaction.Default);  
37 dw_pr.Retrieve();  
38 }  
39  
40 public virtual void cb_clear_Clicked(object sender, EventArgs e) {  
41 Clear();  
42 }  
43  
44 public virtual void dw_pr_Itemchanged(object sender, DataWindowItemCh  
45 string messageText;;  
46  
47 messageText = "itemchanged: ";  
48 messageText += "/ row " + Runtime String(e Row);
```

Rewrite	Node	Location
PBEventDeclarationNode	PBEventDeclarationNode	(1874 to: 1905)

Rule Debugger

Debugging PBForwardPrototypesNode

Bytecode GT

Stack

Transformation

PBForwardPrototypesNode

Generate designer files
process file
[setup post load blocks]
setup post load blocks

self deleteWithWhitespaceBefore: match

Original Source

Run to Cursor

New Source

24 cd_clear cd_clear
25 mle_console mle_console
26 dw_pr dw_pr
27 end type
28 global w_eventhandling w_eventhandling
29
30 type variables
31 string ls_crlf = '~r~n'
32 end variables
33
34 forward prototypes
35 public subroutine newline ()
36 public subroutine clear ()
37 public subroutine log (string logtext)
38 public subroutine logevent ()
39 end prototypes
40
41 public subroutine newline ();

4
3 public partial class WEventhandling : Synchrony.Forms.SpclForm {
4
5
6
7
8
9 public static WEventhandling w_eventhandling;
10
11 public string ls_crlf = "~r~n";
12 forward prototypes
13 public subroutine newline ()
14 public subroutine clear ()
15 public subroutine log (string logtext)
16 public subroutine logevent ()
17 end prototypes
18
19 public subroutine newline ();

Variables

Evaluator

Type	Variable	Value
implicit	self	an a subclass of SmaCCRewriteMatchContext
attribute	continuation	[self performLink: link next on: aSmaCCNode continuation: aBlock]
attribute	match	a PBForwardPrototypesNode(forward prototypes public subroutine newline () public subroutine clear () public subroutine log (string logtext) public subroutine logevent ()
attribute	nodes	a Dictionary [5 items] ('endPrototypesToken' -> {prototypes(870,879,#(14 116))} 'endToken' -> {end(866,868,#(39 116))} 'forwardToken' -> {forward(722,728,#(31 116))} 'logToken' -> {log(866,868,#(39 116))} 'logeventToken' -> {logevent(866,868,#(39 116))})
attribute	rewriteEngine	a SmaCCRewriteEngine
attribute	strings	a Dictionary [5 items] ('endPrototypesToken' -> [cachedString ifNil: [cachedString := self computeStringFor: value]] 'endToken' -> [cachedString ifNil: [cachedString := self computeStringFor: value]] 'forwardToken' -> [cachedString ifNil: [cachedString := self computeStringFor: value]] 'logToken' -> [cachedString ifNil: [cachedString := self computeStringFor: value]] 'logeventToken' -> [cachedString ifNil: [cachedString := self computeStringFor: value]])
implicit	thisContext	a subclass of SmaCCRewriteMatchContext(SmaCCRewriteMatchContext)>>code
implicit	stack top	a PBForwardPrototypesNode(forward prototypes public subroutine newline () public subroutine clear () public subroutine log (string logtext) public subroutine logevent ()

Questions?

<http://www.refactoryworkers.com/SmaCC/>

Download for Pharo:

Gofer new

```
smalltalkhubUser: 'JohnBrant'  
  project: 'SmaCC';  
  configurationOf: 'SmaCC';  
  loadBleedingEdge
```