An AST for Autohighlight

Scott Williams

Scotty Allen

10 Nov 2005

Possible use of predefined computational roles: 8/10. We cannot use predefined computational roles for all our name analysis because it's not syntactically possible to distinguish between defining and applied occurrences of symbol names in the concrete grammar section of the highlighter specification. This is a limitation due to the nature of the BNF grammar that autohighlight inherited from Eli, so further improvement is not possible without breaking compatibility. Additionally, while it is an error to attempt to highlight a literal which doesn't occur in the concrete grammar, our current AST doesn't allow for this to be done with predefined computational roles. Part of the reason for this is that we so far haven't learned to do name analysis on language feature that act like records (for example, struct.member access in C).

Ease of computation: 9/10. We made a decision to combine all the files into one tree, which allows us to do computation across all three of them easily. We used the same symbol names across the scanner, bnf, and autohighlight grammars so that we could simplify computations. However, one thing we weren't able to do is group all the symbols that we require for a lot of the computations in one portion of the tree, and the shape of the AST doesn't reflect the shape of the output file, so we have to do lots of custom output and using ptg's might be hard. For instance, to form a simple regex rule for a symbol for one of the autohighlighter files, we need to get the regex to match the symbol from the gla file portion of the tree, the color mapping from the autohighlighter file portion of the tree, and the surrounding context from the gla portion of the tree. Most of this information will need to be computed because it cannot be inferred from the AST.

Extensibility: 8/10. Capabilities of text editors change over time. New text editors may be supported by autohighlighter in the future. The section of autohighlighter most likely to change because of added functionality is the color specifications, particularly the color criteria such as foreground color and font size). We chose not to encode these in the AST by writing concrete rules for every color criterion (font-weigh, etc), but rather to treat them as identifiers, and handle them using name analysis and type analysis on known keys.

autohighlight.lido[1] ≡

```
RULE: Document ::= '{' GlaFile '}' '{' ConFile '}' '{' AhFile '}' END;

/* Name analysis role extension points */
RULE: SymbolDef ::= Identifier END;
RULE: SymbolUse ::= Identifier END;
RULE: ColorDef ::= Identifier END;
RULE: ColorUse ::= Identifier END;
RULE: PreDefPatternUse ::= Identifier END;

RULE: GlaFile LISTOF Specification END;
RULE: Specification ::= SymbolDef ':' RegularExpression '.' END;
RULE: Specification ::= SymbolDef ':' PreDefPatternUse '.' END;

RULE: ConFile LISTOF Production END;
RULE: Production ::= SymbolDef ':' Elements '.' END;
RULE: Elements LISTOF Element END;
RULE: ConSymbol ::= SymbolUse END;
RULE: ConSymbol ::= Literal END;
RULE: Element ::= ConSymbol END;
RULE: Element ::= '&' ConSymbol END;
RULE: Element ::= '@' ConSymbol END;
```

```
RULE: Element ::= '$' ConSymbol END;
/* These are either literals or symbol names, according to
http://eli-project.sourceforge.net/elionline4.4/syntax_6.html#IDX138 */

RULE: AhFile LISTOF Statement END;
RULE: Statement ::= SyntaxGroupRule END;
RULE: Statement ::= MappingRule END;
RULE: SyntaxGroupRule ::= ColorDef '{' ColorRules '}' END;
RULE: ColorRules LISTOF ColorRule END;
RULE: ColorRule ::= ColorRuleName ':' Value ';' END;
RULE: Value ::= Identifier END;
RULE: Value ::= Literal END;
RULE: MappingRule ::= ColorUse ':' RuleRefs '.' END;
RULE: RuleRefs LISTOF RuleRef END;
RULE: RuleRef ::= SymbolUse END;
RULE: RuleRef ::= SymbolUse '[' Integer ']' END;
RULE: RuleRef ::= SymbolUse '[' Literal ']' END;
RULE: RuleRef ::= SymbolUse '[' Literal ']' '[' Integer ']' END;
RULE: RuleRef ::= Literal END;
```

This macro is attached to a product file.

autohighlight.con[2] ≡

```
ColorRuleName: Identifier / CssIdentifier .
```

This macro is attached to a product file.

autohighlight.gla[3] ≡

```
Identifier: C_IDENTIFIER [mkidn]
CssIdentifier: $[a-zA-Z_][-a-zA-Z_0-9]* [mkidn]
Literal: MODULA2_LITERALSQ [mkstr]
Integer: C_INTEGER [mkidn]
RegularExpression: $\$[^\040]+ [mkidn]
```

This macro is attached to a product file.