Proxima Manual

Martijn M. Schrage

June 23, 2003

1 Startup

The text file "HeliumFile.hs" contains the initial document.

Proxima has two different foci (focus == cursor/selection) that will be integrated in a future version. The *document focus* is shown with a light blue background, and the *presentation focus* is shown with dark blue rectangles. The presentation focus is similar to a cursor or selection in a regular text editor. When text is typed, it is inserted at the presentation focus.

The available features depend on the level in which the document is recognized by the editor. There are three levels: not parsed, parsed, and parsed & type checked. Edit operations on a level may only be performed when the document is recognized at that level. However, in the current prototype, illegal edit operations are not disallowed, but operate on the last version of that level.

2 Not parsed

The only available features in the unparsed document are presentation navigation (**ArrowKeys** and **LeftClick**), presentation focus enlargement (**Shift-ArrowKeys** and **Shift-LeftClick**), and presentation edit operations (typing text, **Ctrl-x**, **Ctrl-c**, and **Ctrl-v**). Presentation navigation is still rather basic: it does not skip invisible empty elements, causing it to sometimes stay in the same place after a left or right move, and it does not handle lines containing elements of varying height well (eg. when a fraction is displayed on a line).

Backspace and **Delete** can be used to delete presentation elements. However, both operations suffer from the same problem as the navigation operations, so sometimes they have to be performed repeatedly before anything happens. Deletion with the mouse is the easiest for now.

3 Parsed

The document can be parsed with **F1**. If the parse fails, the background is colored grey and the parse errors are shown below the source. If the parse succeeds, the following edit commands are available.

Structural navigation:
The document focus can be changed with **Alt-LeftClick**, **Ctrl-Arrowkeys** and <**Select**> from the context menu (**Ctrl-LeftClick**). Downward navigation selects the first child. When **Alt-LeftClick** is overridden by the presentation, <**Select**> from the context menu can be used.

Integrated structural edit: (nice in Helium lists and tuples)
Document editing has its own cut (**Ctrl-d**), copy (**Ctrl-f**) and paste (**Ctrl-g**) operations, which are also available from the context menu (**Ctrl-LeftClick**).

Below the <**Select**> item in the context menu are replace items of the form **"Constructor {child} ..."** that can be used to replace the document structure on which the **Ctrl-LeftClick** was performed. Types appearing in braces are document holes.

In list structures (`Decls`, `Alts`, or `Exps`), **"add"** items are present for inserting after the current item, as well as "<**paste before**>" and "<**paste after**>" items.

Simple value computations:
A naive computation of the value of the function is shown in comment before its definition. It does not work for case statements and functions are shown as `<function>`.

List of variables in scope:
Below the source is a list of variables currently in scope. An **Alt-LeftClick** in the list moves the document focus to the corresponding declaration. If the document is type correct, the types of the variables in scope are also shown in the list.

Jump to declaration
The context menu (**Ctrl-LeftClick:**) of an identifier in an expression has the **"Jump to declaration"** item that sets the document focus to its declaration.

Selective auto layout:
A top-level declaration has the context menu item **"Enable auto-layout"**. When it is turned on, the whitespace is determined by a pretty printer and not freely editable anymore. When auto-layout is turned off, the layout of the pretty printed function can be adapted by normal presentation editing (typing and removing spaces and newlines).

Collapsed functions:
Anywhere in declaration, the context menu has a **"Collapse: name"** item that collapses the definition. Collapse may also be performed on declarations in let expressions. A collapsed definition can be expanded by performing an **Alt-LeftClick** on the "[...]", or by selecting **"Expand"** from the context menu.

In the current prototype, the layout is lost if a collapsed function is parsed. To remedy this situation: turn on autolayout for the function.

4 Parsed & type checked

The document is parsed and type checked after **F2** has been pressed. Type checked does not necessarily mean type correct.

Errors displayed in source:
Type errors appear below the program source. Squiglies in the source show the location of errors.

Jump to error:
An **Alt-LeftClick** on an error message sets the document focus to the location of the error in the source.

Derived type signatures:
Derived type signatures appear before each function definition. The derived signatures are not editable.

Type of focused item:
The type of the program construct that is in document focus is shown at the top of the screen. When the type does not appear, the focus may be too far down in the tree (eg. on an `Ident` instead of an `IdentExp`). Moving the focus up will show the type. Declarations have no type.

5 Chess board

Type "`Chess: board`" (no semi-colon) between two top-level declarations to insert a chess board (or use the context menu on a declaration to either replace the declaration by a chess board or **add** a chess board right behind it, note that in the latter case no whitespace is provided). Document selection of a piece (**Alt-LeftClick**) shows its possible moves. **Alt-LeftClick** on a possible move location performs the move. Normal document edit operations such as cut (**Ctrl-d**) and paste (**Ctrl-g**) are also available. The context menu (**Ctrl-LeftClick**) may be used to create new pieces.

***** WARNING ***:** Do not put pawns on the back row! The chess move generator can't handle this case and will cause the entire editor to crash.

6 Overview of edit commands

Arrangement editing:	
Zoom in:	**Alt-=**
Zoom out:	**Alt--** (zooming out to far causes a crash)
Debug view:	**F9** (shows what's really going on)
Presentation editing:	
navigate:	**ArrowKeys**
	LeftClick
Extend focus:	**Shift-ArrowKeys**
	Shift-LeftClick
copy:	**Ctrl-c**
cut:	**Ctrl-x**
paste:	**Ctrl-v**
Doc editing:	
Select:	**Alt-LeftClick** (may be overridden)
Navigate:	**Ctrl-ArrowKeys**
Context menu:	**Ctrl-LeftClick**
Copy:	**Ctrl-d**
Cut:	**Ctrl-f**
Paste:	**Ctrl-g**
Other:	
Parse:	**F1**
Type check:	**F2**
Quit:	**Ctrl-q**

The Save option in the file menu saves a simple XML representation of the edited document.

7 Document Type and Concrete Syntax

```
data Document = RootDoc Decls          -- Decls


data Decls = ConsDecls Decl Decls      -- Decl Decls
           | NilDecls                  --


data Decl = Decl Ident Exp             -- Ident "=" Exp ";"
          | BoardDecl Board            -- "Chess:" Board

data Ident = Ident String              -- String
```

```
data Exps = ConsExps Exp Exps          -- Exp "," Exps  {not entirely: last has no ","}
          | NilExps                    --

data Alts = ConsAlts Alt Alts          -- Alt Alts
          | NilAlts                    --


data Exp = PlusExp    Exp   Exp        -- Exp "+" Exp
         | TimesExp   Exp   Exp        -- Exp "*" Exp
         | DivExp     Exp   Exp        -- Exp "%" Exp
         | PowerExp   Exp   Exp        -- Exp "^" Exp
         | BoolExp    Bool             -- "True" | "False"
         | IntExp     Int              -- "0" | "1" | "2" | ...
         | LamExpent  Exp              -- "\" Exp "->" Exp
         | AppExp     Exp   Exp        -- Exp Exp
         | CaseExp    Exp   Alts       -- "case" Exp "of" Alts
         | LetExp     Decls Exp        -- "let" Decls "in" Exp
         | IdentExp   Ident            -- Ident
         | IfExp      Exp   Exp Exp    -- "if" Exp "then" Exp "else" Exp
         | ParenExp   Exp              -- "(" Exp ")"
         | ListExp    Exps             -- "[" Exps "]"
         | ProductExp Exps             -- "(" Exps ")"

data Alt = Alt Ident Exp               -- Ident "->" Exp ";"


data Board       = Board BoardRow BoardRow BoardRow BoardRow
                   BoardRow BoardRow BoardRow BoardRow         -- "board"

data BoardRow    = BoardRow BoardSquare BoardSquare BoardSquare BoardSquare
                            BoardSquare BoardSquare BoardSquare BoardSquare

data BoardSquare = King   Bool | Queen Bool | Bishop Bool |
                   Knight Bool | Rook  Bool | Pawn   Bool | Empty
```

8 Bugs and issues

Some known bugs and problems of the editor are:

- Focus move doesn't skip presentation positions that have the same arrangement position.
- Focus gets lost often after parsing or structural edit
- Computations may not be recursive because this loops the simple evaluator (and Proxima)
- The right mouse button is not supported by ObjectIO (ObjectIO is the library used for rendering)
- Screen flickering cannot be fixed in ObjectIO without loss of speed.
- Parse errors create a newline at end of declarations
- Poor display of parse errors.
- HoleDecls cannot be parsed.

- Integration between edit operations on different levels is not automatic yet
- The editor is rather slow

New rendering and parsing modules, together with an improved focus model and basic top level incrementality will solve most of these problems.