Arbejdsblade for vejledermøde 27.02.13

Gruppe d402f13

Tirsdag 26 februar, 2013

Her er en kort opsummering af arbejdet siden sidste møde til vejledermødet d. 27.02.13.

1 Syntaks

Her vises et lille eksempel på Tic Tac Toe implementeret i vores sprog, så du har en fornemmelse af syntaksen. Det ses her:

```
game {
  title "Noughts and Crosses"
  players [Noughts Crosses]
  turnOrder [Crosses Noughts]
  board {
    grid {
      width 3
      height 3
    }
  }
  piece {
    name "XOPiece"
    possibleDrops (emptySquares[board])
  }
  winCondition (
    notEmpty[findSquares[/friend n friend n friend/]]
      or notEmpty[findSquares[/friend e friend e friend/]]
      or notEmpty[findSquares[/friend nw friend nw friend/]]
      or notEmpty[findSquares[/friend ne friend ne friend/]]
  )
  drawCondition (
    isFull[board]
  )
}
```

2 Grammar

Vi er begyndt at specificere en formel grammatik til sproget, da vi endelig er kommet på en syntaks, der giver mening for alle gruppemedlemmer. Dette arbejdes på i løbet af i dag (26/02/13), så derfor kan der godt være nogle finurligheder (fx vores egen version af EBNF). Vi begynder at dokumentere syntaksen i rapporten i løbet af denne uge. Udkastet ses her:

Character classes

```
decimal              -> “0” | “1” | ... | “9”
lowercase            -> “a” | “b” | ... | “z”
uppercase            -> “A” | “B” | ... | “Z”
anycase              -> lowercase | uppercase
unichar              -> any unicode character
strchar              -> unichar except for “\\” and “\””
```

Reserved

```
keyword              -> “game” | “piece” | “this” | “width” | ”height”
                      | “title” | “players” | “turnOrder” | “board”
                      | “grid” | “setup” | “wall” | “name” | “possibleDrops”
                      | “possibleMoves” | “winCondition” | “tieCondition”
operator             -> “and” | “or”
pattern_keyword      -> “friend” | “foe” | “this” | “empty”
pattern_operator     -> “*” | “?” | “+” | “!”
```

Literals

```
integer              -> decimal{decimal}
direction            -> “n” | “s” | “e” | “w” | “ne” | “nw”
                      | “se” | “sw”
coordinate           -> uppercase{uppercase}decimal{decimal}
string               -> “\”” {strchar | “\\” unichar} “\””
```

Identifiers

```
function             -> lowercase anycase{anycase}
identifier           -> uppercase {anycase}
variable             -> “$” anycase {anycase}
```

Program structure

```
program              -> {function_def} game_decl
function_def         -> “define” function “[“ {variable} “]” expression
game_decl            -> “game” declaration_struct
declaration_struct   -> “{“ declaration {declaration} “}”
declaration          -> (keyword | identifier) structure
structure            -> declaration_struct | expression
```

Expressions

```
expression           -> function_call
                      | element operator expression
                      | if_expr
                      | lambda_expr
                      | element
element                    -> “(“ expression “)
                      | variable
                      | list
                      | pattern
```

```
                    | keyword
                    | direction
                    | coordinate
                    | integer
                    | string
                    | identifier
function_call       -> function list
if_expr             -> “if” expression “then” expression “else” expression
lambda_expr         -> “#” {varlist} “=>” expression
list                -> “[“ {element} “]”
```

Patterns

```
pattern             -> “/” pattern_expr “/”
pattern_expr        -> pattern_expr {pattern_expr}*
                    | ( pattern_expr )
                    | ( pattern_expr ) integer
                    | pattern_expr "*"
                    | pattern_expr "?"
                    | pattern_expr "+"
                    | pattern_val
pattern_val         -> direction
                    | pattern_check
                    | “!” pattern_check
pattern_check       -> “friend”
                    | “foe”
                    | “empty”
                    | “this”
                    | id
```

3 Scanner

Der er også blevet lavet første fungerende udkast til en scanner ud fra de tokens, vi har har defineret i syntaksen. Vi er nu gået i gang med at dokumentere scanneren i rapporten.

4 Parser

Vi er lige begyndt på parseren. Meningen er, at den nuværende version inden for dette sprint "kun" skal kunne parse nogle terminals, vi ved skal være der, da vi er stadig lidt usikre på nogle non-terminals.