1 Preamble

- section name and its parents
- basic process header
- typed channel
- generic typed channels
- synchronisation channels
- various auxiliary declarations used
- WHAT KIND OF TEST?

$\textbf{section}\ \ \mathit{action\_grammar\_rules}\ \textbf{parents}\ \ \mathit{circus\_toolkit}$

$|\ \ \mathit{outside} : \mathbb{N}$

$\textbf{channel}\ \ c : \mathbb{N} \times \mathbb{N} \times \mathbb{N} \times \mathbb{N}$

$\textbf{process}\ \ \mathit{CircusTimeActionTests} \mathrel{\widehat=} \textbf{begin}$

$|\ \ n1, n2 : \mathbb{N}$
$|\ \ x?, y!, z? : \mathbb{N}$

$|\ \ f : \mathbb{N} \rightarrow \mathbb{N} \times \mathbb{N}$

$S == [\, y : \mathbb{N} \,]$

$A \mathrel{\widehat=} \textbf{Skip}$

2 Example 1 — *CIRCWAIT*

Production rule:

`CIRCWAIT:cw expression:e`

- simple wait with expressions of various kinds and no following action
- expressions can come from within and outside the process
- expressions that look **funny like partially applied function or theta**

Shouldn't $Test2/3$ fail to typecheck somehow because their types too generous? I guess these (wicked) expressions are for the typechecker. Also, what about parenthesis? Are they needed / required here?

$$Test0 \mathrel{\widehat=} \mathbf{wait}\, 10 + outside$$
$$Test1 \mathrel{\widehat=} \mathbf{wait}\, n1 + n2$$
$$Test2 \mathrel{\widehat=} \mathbf{wait}\, f$$
$$Test3 \mathrel{\widehat=} \mathbf{wait}\, \theta\, S$$
$$Test4 \mathrel{\widehat=} \mathbf{wait}[\, x : \mathbb{N} \mid x > 10\,]$$

We would also like to add some examples with erroneous productions like

`\t1 TestX \circdef \circwait 10 \then \Skip`

That is, one might get this (or others) as common mistakes because of the other type (in next section)? That means adding spurious grammar productions for common (error) patterns will help give better error messages.

Production rule:

`CIRCWAIT DECLWORD:dw COLON expression:e CIRCSPOT circusAction:ac`

- complex wait patterns with ensuing actions
- complex wait patterns with duplicated names (i.e. already declared)
- complex wait patterns with possibly unacceptable names?

Here some tests reuse declared (global) names, which should be caught by the typechecker yet accepted by the parser given the production expects a DECLWORD, which will also accept strokes (?, !, etc), that you might want to avoid as well?

$$Test5 \mathrel{\widehat=} \mathbf{wait}\, x : 10 + outside \bullet \mathbf{Skip}$$
$$Test6 \mathrel{\widehat=} \mathbf{wait}\, y : n1 + n2 \bullet \mathbf{Skip}$$
$$Test7 \mathrel{\widehat=} \mathbf{wait}\, z : f \bullet \mathbf{Skip}$$
$$Test8 \mathrel{\widehat=} \mathbf{wait}\, w : \theta\, S \bullet \mathbf{Skip}$$
$$Test9 \mathrel{\widehat=} \mathbf{wait}\, x? : S \bullet \mathbf{Skip}$$
$$Test10 \mathrel{\widehat=} \mathbf{wait}\, f : S \bullet \mathbf{Skip}$$

3 Example 2 — *CIRCSTARTBY* and *CIRCENDBY*

Production rule:

```
LCIRCTIME expression:e RCIRCTIME CIRCSTARTBY circusAction:ca
circusAction:ca CIRCENDBY LCIRCTIME expression:e RCIRCTIME
```

Again same issues about expressions apply.

$Test11 \mathrel{\widehat=} (10 + outside)\ startby\ \mathbf{Skip}$

$Test12 \mathrel{\widehat=} (n1 + n2)\ startby\ \mathbf{Skip}$

$Test13 \mathrel{\widehat=} (f)\ startby\ \mathbf{Skip}$

$Test14 \mathrel{\widehat=} (\theta\, S)\ startby\ \mathbf{Skip}$

$Test15 \mathrel{\widehat=} (x?)\ startby\ \mathbf{Skip}$

$Test16 \mathrel{\widehat=} A\ endby\ (10 + outside)$

$Test17 \mathrel{\widehat=} A\ endby\ (n1 + n2)$

$Test18 \mathrel{\widehat=} A\ endby\ (f)$

$Test19 \mathrel{\widehat=} A\ endby\ (\theta\, S)$

$Test20 \mathrel{\widehat=} A\ endby\ (x?)$

4 Example 3 — *CIRCTIMEOUT*

Production rule:

```
circusAction:al CIRCTIMEOUT LCIRCTIME expression:e RCIRCTIME  circusAction:ar
```

$Test21 \mathrel{\widehat=} A\ timeout\ (10 + outside) A$

$Test22 \mathrel{\widehat=} A\ endby\ (f) A\ ;\ \ A$

5 Example 4 — *CIRCTIMEDINTERRUPT*

Production rule:

```
circusAction CIRCTIMEDINTERRUPT LCIRCTIME expression RCIRCTIME circusAction
```

$Test23 \mathrel{\widehat=} A\ interrupt\ (10 + outside) A$

$Test24 \mathrel{\widehat=} A\ interrupt\ (f) A\ ;\ \ A$

6 Example 5 — timed prefix

Production rule:

```
communication PREFIXTHEN:pt LCIRCTIME expression RCIRCTIME circusAction
```

Given complexity of communication and channels (i.e. smart scanning), for timed prefix, we'd better have thorough tests. I will copy those from dot-field-singleenv.tex as they already are thorough for communication and expand/extend them here for timed communication.

$$Test25 \mathrel{\widehat=} c?x?y?z \longrightarrow (10 + outside)\mathbf{Skip}$$

$$Test26 \mathrel{\widehat=} c?x!n1.n2 \longrightarrow (x?)\mathbf{Skip}$$

$$Test27 \mathrel{\widehat=} c!n1?x.n2 \longrightarrow (f)\mathbf{Skip}$$

$$Test28 \mathrel{\widehat=} c.n1!n2?x \longrightarrow (20)\,d \longrightarrow (\theta\, S)\mathbf{Skip}$$

$$Test29 \mathrel{\widehat=} d \longrightarrow (f)\,e \longrightarrow (f)\mathbf{Skip}$$

$$Test30 \mathrel{\widehat=} c\,?i\,!(f\ i) \longrightarrow (f)\mathbf{Skip}$$

$$Test31 \mathrel{\widehat=} c.(S.y)?z \longrightarrow (20)\mathbf{Skip}$$

$$Test32 \mathrel{\widehat=} c.(x?)!(y!)!(z?) \longrightarrow (20)\mathbf{Skip}$$

* $S \in \mathbb{P}(\langle\!|y == \mathbb{N}|\!\rangle)$**, hence** $S.y \in \mathbb{N}$**.**

* $x?, y!, z?$ **are decorated names; usually they appear in schemas.** If this proves too complicated, perhaps having it separate as another file is a good away to divide-and-conquer.

Description

Action	Communication pattern
$Test25$	In($x, \mathbb{N}$), In($y, \mathbb{N}$), In($z, \mathbb{N}$)
$Test26$	In($x, \mathbb{N}$), Out($n1$), Dot($n2$)
$Test27$	Out($n1$), In($x, \mathbb{N}$), Dot($n2$)
$Test28$	Dot($n1$), Out($n2$), In($x, \mathbb{N}$), Synch
$Test29$	Synch, Synch, Synch
$Test30$	In($i, \mathbb{N}$), Out($\mathbb{N} \times \mathbb{N}$)
$Test31$	Dot($S.y$), In($z, \mathbb{N} \times \mathbb{N}$)
$Test32$	Out($x?$), Out($y!$), Out($z?$)

QUESTION: what about prefix with restricting expression and time? The grammar has no production for it, and yet would seem sensible to have them for completion? Here are how they are in *Circus*.

$$Test33 \mathrel{\widehat=} c?x : (x > 1)!(f\ x) \longrightarrow \mathbf{Skip}$$

$$\begin{aligned}Test34 \mathrel{\widehat=} {} & c?x\longrightarrow \\ & \quad c?z : (z > x.1).(f\ (x.2 + x.3)) \longrightarrow \mathbf{Skip}\end{aligned}$$

Description

Action	Communication pattern
$Test33$	$\mathrm{In}(x, \{\, v : \mathbb{N} \mid v > 1 \,\})$, $\mathrm{Out}(\mathbb{N} \times \mathbb{N})$
$Test34$	$\mathrm{In}(x, \mathbb{N} \times \mathbb{N} \times \mathbb{N})$; $\mathrm{In}(z, \{\, v : \mathbb{N} \mid v > x.1 \,\})$, $\mathrm{Out}(\mathbb{N} \times \mathbb{N})$

*** type on inputs are restricted according to given predicate.**
*** $Test34$ input on z is from "$?z : (z > x.1)$".**
Finally we have generic channels version with time

$$Test34 \mathrel{\widehat=} g[\mathbb{N} \times \mathbb{N} \times \mathbb{N}]?x!n1.n2 \longrightarrow (20)\mathbf{Skip}$$

$$Test35 \mathrel{\widehat=} g.n1.(f\ n1) \longrightarrow (20)\mathbf{Skip}$$

7 Example 6 — AT timed prefix

Production rule:

```
communication ATTIME DECORWORD  PREFIXTHEN  circusAction
communication ATTIME DECORWORD PREFIXTHEN LCIRCTIME expression RCIRCTIME circusAction
```

Here I am adding names for the *at* symbol that are either repeated or the "wrong" type.

$$Test36 \mathrel{\widehat=} c?x?y?z\,at\,X \longrightarrow (10 + outside)\mathbf{Skip}$$

$$Test37 \mathrel{\widehat=} c?x!n1.n2\,at\,Z \longrightarrow (x?)\mathbf{Skip}$$

$$Test38 \mathrel{\widehat=} c!n1?x.n2\,at\,Y \longrightarrow (f)\mathbf{Skip}$$

$$Test39 \mathrel{\widehat=} c.n1!n2?x\,at\,x? \longrightarrow (20)\,d\,at\,f \longrightarrow (\theta\, S)\mathbf{Skip}$$

$$Test40 \mathrel{\widehat=} d \longrightarrow (f)\,e \longrightarrow (f)\mathbf{Skip}$$

$$Test41 \mathrel{\widehat=} c\,?i\,!(f\ i)\,at\,W \longrightarrow \mathbf{Skip}$$

$$Test42 \mathrel{\widehat=} c.(S.y)?z\,at\,K \longrightarrow \mathbf{Skip}$$

$$Test43 \mathrel{\widehat=} c.(x?)!(y!)!(z?)\,at\,Q \longrightarrow (20)\mathbf{Skip}$$

8 Example 6 — Processes TODO

Production rule:

```
process:pl CIRCENDBY LCIRCTIME expression:e RCIRCTIME
LCIRCTIME expression:e RCIRCTIME CIRCSTARTBY process:pr
process:pl CIRCTIMEOUT  LCIRCTIME expression:e RCIRCTIME  process:pr
process:pl CIRCTIMEDINTERRUPT:ti  LCIRCTIME expression:e RCIRCTIME  process:pr
```

9 (! Prolegomena) — basic process footer

It just terminates

- **Skip**

end

10 LaTeX

You also need to update circus-time.sty to contain something akin to

```
% TODO: These commands also ought to become part of something like circus-time.sty
%  In particular the selection of the appropriate UNICODE character from within
% the font's symbol table
\makeatletter
\newcommand{\circwait}{\zpreop{\circuskeyword{wait}}}
\newcommand{\circstartby}{\zbinop{startby}}
\newcommand{\circendby}{\zbinop{endby}}
\newcommand{\lcirctime}{\zopenop{(}}
\newcommand{\rcirctime}{\zcloseop{)}}
\newcommand{\circtimeout}{\zbinop{timeout}}
\newcommand{\circtimedinterrupt}{\zbinop{interrupt}}
\newcommand{\circat}{\zordop{at}}
\makeatother
```

Notice that the choices I've made quickly here don't make for pretty latex! You need to fix / arrange those by looking at how it's done properly for Z and Circus.