Lando System Specification Language v2 Grammar

September 9, 2024

1 Notation

The grammar for the Lando System Specification Sublanguage is written in the EBNF notation. The main elements of the notation that we utilize are:

- Keyword terminals are represented in `typewriter` font.
- Single-character terminals appear in single quotes, e.g. `'!'` or `'\n'`.
- More complex terminals are represented using standard regular expressions written as «`regexp`», which may include escaped control characters such as `\n`. Regular expressions may use character classes (with negation). Note that the pattern «`/`» matches the literal character forward slash.
- An optional *thing* is written as [*thing*].
- Zero or more repetitions of *thing* are written as { *thing* }.
- One or more repetitions of *thing* are written as { *thing* }$^+$.
- If S and T are non-terminals or regular expressions, $S - T$ denotes the set of strings matching S but not T.
- *EOF* represents end-of-file; it can be thought of as a special terminal that is not consumed, but rather is replicated as often as necessary to match the grammar when parsing.
- Whitespace and line separators are significant throughout. Note that the line separator terminal $\mathcal{N}$ consumes any whitespace immediately following the actual end-of-line character(s). Two consecutive line separators (i.e. matching $\mathcal{N}\mathcal{N}$) are always lexed instead as a single blank line terminal $\mathcal{B}$; this distinction is necessary to resolve certain ambiguities in the grammar for paragraphs. The general line separator $\mathcal{L}$ is the union of $\mathcal{N}$ and $\mathcal{B}$.
- In cases of ambiguity, productions involving repetition behave "greedily," i.e. they match as much of the input as possible. For example, because the Sentence Body production uses { }, the string "`abc def ghi`" parses as a single sentence containing three words, rather than as three sentences of one word each.

1.1 Context-Free Grammar

landoSource	::=	$\{\,\mathcal{L}\,\}\,\{\,body\,\}\,\{\,\mathcal{L}\,\}\,\{\,\mathcal{CL}\,\}\,\{\,\mathcal{L}\,\}\;EOF$	*Lando source*
specElement	::=	*system* | *subsystem* | *subsystemImport* | *component* | *componentImport* | *events* | *scenarios* | *requirements* | *relation*	*Top-level element*
system	::=	$\{\,\mathcal{CL}\,\}$ `system` *name* $[\,abbrev\,]\,[\,\mathcal{C}\,]\,\{\,\mathcal{L}\,\}^+$ *explanation* $\{\,\mathcal{CL}\,\}\,[\,indexing\;blockend\,]$ $[$ `contains` $\{\,\mathcal{L}\,\}$ *body* `end` $[\,\mathcal{C}\,]\;blockend\,]$	*System*
body	::=	$\{$ *specElement* $\}$	*Source/(Sub)system body*
subsystem	::=	$\{\,\mathcal{CL}\,\}$ `subsystem` *name* $[\,abbrev\,]\,\{\,clientClause\,\}\,[\,\mathcal{C}\,]\,\{\,\mathcal{L}\,\}^+$ *explanation* $\{\,\mathcal{CL}\,\}\,[\,indexing\;blockend\,]$ $[$ `contains` $\{\,\mathcal{L}\,\}$ *body* `end` $[\,\mathcal{C}\,]\;blockend\,]$	*Subsystem*
subsystemImport	::=	$\{\,\mathcal{CL}\,\}$ `import` $\{\,\mathcal{S}\,\}^+$ `subsystem` *name* $[\,abbrev\,]\,\{\,clientClause\,\}\,[\,\mathcal{C}\,]\;blockend$	*Subsystem import*
explanation	::=	*paragraph*	*Explanation*
abbrev	::=	$\{\,\mathcal{S}\,\}$ `'('` $\{\,\mathcal{S}\,\}$ *nameWord* $\{\,\mathcal{S}\,\}$ `')'` $\{\,\mathcal{S}\,\}$	*Abbreviation*
inheritClause	::=	`inherit` *qname* $\{\,\mathcal{L}\,\}\,\{$ `','` $\{\,\mathcal{L}\,\}$ *qname* $\{\,\mathcal{L}\,\}\,\}$	*Inherit clause*
clientClause	::=	`client` *qname* $\{\,\mathcal{L}\,\}\,\{$ `','` $\{\,\mathcal{L}\,\}$ *qname* $\{\,\mathcal{L}\,\}\,\}$	*Client clause*
indexing	::=	`indexing` $\{\,\mathcal{S}\,\}\,\{\,\{\,\mathcal{L}\,\}^+$ *indexEntry* $\}$	*Indexing*
indexEntry	::=	*indexValue* `':'` *indexValuePart* $\{\,\{\,\mathcal{L}\,\}^+$ *indexValuePart* $\}$	*Index Entry*
indexValuePart	::=	*indexValue* $\{\,\mathcal{L}\,\}\,[\,\mathcal{C}\,]$	*Index Value Part*
component	::=	$\{\,\mathcal{CL}\,\}$ `component` *name* $[\,abbrev\,]\,\{\,inheritClause$ | *clientClause* $\}\,[\,\mathcal{C}\,]\,\{\,\mathcal{L}\,\}^+$ *explanation* $\{$ *componentFeature blockend* $\}$	*Component definition*
componentImport	::=	$\{\,\mathcal{CL}\,\}$ `import` $\{\,\mathcal{S}\,\}^+$ `component` *name* $[\,abbrev\,]\,\{\,clientClause\,\}\,[\,\mathcal{C}\,]\;blockend$	*Component import*
componentFeature	::=	*constraint* | *command* | *query*	*Component Feature*
constraint	::=	$\{\,\mathcal{CL}\,\}$ *sentBody* `'.'` $[\,wordSep\,]\,[\,\mathcal{C}\,]$	*Constraint Feature*
query	::=	$\{\,\mathcal{CL}\,\}$ *sentBody* `'?'` $[\,wordSep\,]\,[\,\mathcal{C}\,]$	*Query Feature*
command	::=	$\{\,\mathcal{CL}\,\}$ *sentBody* `'!'` $[\,wordSep\,]\,[\,\mathcal{C}\,]$	*Command Feature*
events	::=	$\{\,\mathcal{CL}\,\}$ `events` *name* $[\,\mathcal{C}\,]\,\{\,\mathcal{L}\,\}^+\,\{$ *eventEntry* $\}$	*Events*
eventEntry	::=	$\{\,\mathcal{CL}\,\}$ *name* $[\,\mathcal{C}\,]\,\{\,\mathcal{L}\,\}^+$ *paragraph*	*Event Entry*
scenarios	::=	$\{\,\mathcal{CL}\,\}$ `scenarios` *name* $[\,\mathcal{C}\,]\,\{\,\mathcal{L}\,\}^+\,\{$ *scenarioEntry* $\}$	*Scenarios*
scenarioEntry	::=	$\{\,\mathcal{CL}\,\}$ *name* $[\,\mathcal{C}\,]\,\{\,\mathcal{L}\,\}^+$ *paragraph*	*Scenario Entry*
requirements	::=	$\{\,\mathcal{CL}\,\}$ `requirements` *name* $[\,\mathcal{C}\,]\,\{\,\mathcal{L}\,\}^+\,\{$ *requirementEntry* $\}$	*Requirements*
requirementEntry	::=	$\{\,\mathcal{CL}\,\}$ *name* $[\,\mathcal{C}\,]\,\{\,\mathcal{L}\,\}^+$ *paragraph*	*Requirement Entry*
relation	::=	$\{\,\mathcal{CL}\,\}$ `relation` *qname* $\{\,inheritClause$ | *clientClause* $\}^+\,[\,\mathcal{C}\,]\;blockend$	*Relation Declaration*
qname	::=	*name* | *qname* `':'` *name*	*Name*
name	::=	$\{\,\mathcal{S}\,\}$ *nameWord* $\{\,\{\,\mathcal{S}\,\}^+$ *nameWord* $\}\,\{\,\mathcal{S}\,\}$	*Name facet*
nameWord	::=	«`[^\r\n \t():]*[^\r\n \t():,.!?]`» − «`//.*`» − *nonName*	*Name word*
nonName	::=	`client` | `inherit`	*Keywords*
indexValue	::=	$\{\,\mathcal{S}\,\}$ *indexValueWord* $\{\,\{\,\mathcal{S}\,\}^+$ *indexValueWord* $\}\,\{\,\mathcal{S}\,\}$	*Index value*
indexValueWord	::=	«`[^\r\n \t]*[^\r\n \t:]`» − «`//.*`»	*Index value word*
paragraph	::=	$\{$ *sentence* $\}^+$ *parend*	*Paragraph*
parend	::=	$[\,\mathcal{C}\,](\mathcal{B}\,\{\,\mathcal{L}\,\}$ | *EOF*$)$	*Paragraph end*
sentence	::=	*sentBody* $[$ *sentTerm* $]\,[$ *wordSep* $]$	*Sentence*
sentBody	::=	*sentWord* $\{$ *wordSep sentWord* $\}\,[$ *wordSep* $]$	*Sentence Body*
wordSep	::=	$\{\,\mathcal{S}\,\}^+$ | $\{\,\mathcal{S}\,\}\,[\,\mathcal{C}\,]\,\mathcal{L}$	*Word Separator*
sentWord	::=	«`[^\r\n \t]*[^\r\n \t.!?]`» − «`//.*`»	*Sentence word*
sentTerm	::=	`'.'` | `'!'` | `'?'`	*Sentence terminator*
blockend	::=	$\{\,\mathcal{L}\,\}^+$ | *EOF*	*Block end*
$\mathcal{S}$	::=	«`[ \t]`»	*Whitespace character*
$\mathcal{N}$	::=	$[$ `'\r'` $]$ `'\n'` $[\,\mathcal{S}\,]$ | `'\r'` $[\,\mathcal{S}\,]$ | *EOF*	*Single line separator*

$\mathcal{B}$ ::= $\mathcal{N}\mathcal{N}$ *Blank line*

$\mathcal{L}$ ::= $\mathcal{N} \mid \mathcal{B}$ *General line seperator*

$\mathcal{C}$ ::= «`//[^\r\n]*`» *Comment*

2 Context-Sensitive Considerations

The following extra-grammatical rules are applied during or after parsing the concrete grammar above.

- If a *system* or *subsystem* element does not have an explicit body, then it has an implicit body that begins immediately after the element and extends until just before the next `system`, `subsystem` or unmatched `end`.
- *subystemImport* an only appear in an (explicit or implicit) *system* or *subsystem* body. *component* and *componentImport* can only appear in an (explicit or implicit) *subsystem* body.
- A *sentence* that does not end with a `'.'`,`'!'`, or `'?'` symbol (possibly followed by white space characters or line separators) induces a warning message.
- Note that because of the "greedy" resolution of ambiguity, this warning can only occur at the end of a paragraph.