SubjectJ BNF Grammars

The following are the BNF grammars for the files that SubjectJ will use to perform the compositions. Note that the project is in the *very* pre-alpha stages, and the grammar are not just subject to, but very likely to change.

The include (*.sji*) file   This file contains the list of all packages and/or class files that SubjectJ will work with. Any package or class file not listed in this file will not be modified or otherwise touched by SubjectJ. This file is important because even if some of the included classes may not be mentioned in any mappings, SubjectJ will update every class named in this file as necessary so that they correctly interact with any newly composed class(es).

$\langle Include \rangle$ ::= $\langle List \rangle$ $\langle Include \rangle$
       | $\langle List \rangle$

$\langle List \rangle$ ::= package $\langle PathList \rangle$ $\langle OptionList \rangle$
       | class $\langle PathList \rangle$

$\langle PathList \rangle$ ::= $\langle Path \rangle$, $\langle PathList \rangle$
       | $\langle Path \rangle$

$\langle Path \rangle$ ::= $\langle String \rangle$. $\langle Path \rangle$
       | $\langle String \rangle$

$\langle OptionList \rangle$ ::= $\langle Option \rangle$ $\langle OptionList \rangle$
       | $\langle Option \rangle$

$\langle Option \rangle$ ::= exclude $\langle ClassSpecification \rangle$
       | recursive

$\langle String \rangle$ ::= c $\langle String \rangle$
       | c

The concern (*.sjc*) file (optional)   This file allows the user to group individual packages or class files into 'mappings' to make referring to them easier when defining subjects. In other words, this is the concern mapping file. This file is optional, however; SubjectJ, like Hyper/J, creates a mapping for every included class file (see the includes file above) – the name of the concern will be the Java fully-qualified name. This name can be used in the subjects file (see below) instead if desired, skipping any need for this file.

$\langle Concerns \rangle$ ::= $\langle Concern \rangle$ $\langle Concerns \rangle$
       | $\langle Concern \rangle$

$\langle Concern \rangle$ ::= package $\langle KeyValue \rangle$
       | class $\langle KeyValue \rangle$
       | interface $\langle KeyValue \rangle$
       | method $\langle KeyValue \rangle$
       | field $\langle KeyValue \rangle$

$\langle KeyValue \rangle$ ::= $\langle Path \rangle$: $\langle String \rangle$

$\langle Path \rangle$ ::= $\langle String \rangle$. $\langle Path \rangle$
       | $\langle String \rangle$

$\langle String \rangle$ ::= c $\langle String \rangle$
       | c

The subject (*.sj*) file   This is where the action happens; the actual subjects are defined in this file. This grammar is the most likely to change based on requirements and feedback as development continues. As it stands currently, there can be one or more subject files – one for each subjective view of the code. The name on top of the file must be distinct among all subjects – unlike Hyper/J, the names are not specified as commandline arguments; the names from the subject files will be used as the names of the folders in which the composed files will be placed. As noted above, the concerns in this file may be the concerns named in the concerns file (above), or they may simply be Java fully-qualified names. SubjectJ will only perform compositions on the named concerns in this file; all other concerns will be untouched with the exception of being updated to work correctly with the newly composed files.

⟨*Subject*⟩   ::= ⟨*Name*⟩ ⟨*Concerns*⟩ ⟨*GeneralStrategy*⟩ ⟨*SpecificStrategies*⟩

⟨*Name*⟩   ::= ⟨*String*⟩

⟨*Concerns*⟩   ::= ⟨*Concern*⟩ ⟨*Concerns*⟩
                 | ⟨*Concern*⟩

⟨*Concern*⟩   ::= ⟨*Path*⟩
                 | ⟨*String*⟩

⟨*GeneralStrategy*⟩ ::= merge by name
                 | noncorresponding merge
                 | override by name

⟨*SpecificStrategies*⟩ ::= ⟨*Strategy*⟩ ⟨*SpecificStrategies*⟩
                 | ⟨*Strategy*⟩

⟨*Strategy*⟩   ::= ⟨*Merge*⟩
                 | ⟨*Override*⟩
                 | . . .

⟨*Merge*⟩   ::= merge ⟨*Method*⟩ with ⟨*Method*⟩
                 | merge ⟨*Method*⟩ with ⟨*Method*⟩ via ⟨*SummaryMethod*⟩
                 | merge ⟨*Field*⟩ with ⟨*Field*⟩ via ⟨*SummaryMethod*⟩

⟨*Override*⟩   ::= override ⟨*Method*⟩ with ⟨*Method*⟩
                 | override ⟨*Field*⟩ with ⟨*Field*⟩

⟨*Class*⟩   ::= ⟨*Concern*⟩ . ⟨*String*⟩
                 | ⟨*Concern*⟩

⟨*Method*⟩   ::= ⟨*Concern*⟩ . ⟨*String*⟩
                 | ⟨*Concern*⟩

⟨*Field*⟩   ::= ⟨*Concern*⟩ . ⟨*String*⟩
                 | ⟨*Concern*⟩

⟨*Path*⟩   ::= ⟨*String*⟩ . ⟨*Path*⟩
                 | ⟨*String*⟩

⟨*String*⟩   ::= c ⟨*String*⟩
                 | c