Compiler API: Mapping

Mapping is the process of generating a UML activity model from the abstract syntax representation of some Alf code. If this mapping is successful, then the Alf compiler updates the context element using the generated elements. How this update is done depends on what the context element is:

- If the context element is an Activity, the nodes and edges of the Activity are replaced with those generated by the compilation.
- If the context element is an Opaque Behavior or Opaque Action, a *CompiledRepresentation* stereotype is applied with its *behavior* tag pointing to the compiled Behavior.
- If the context element is an Opaque Expression, its *behavior* property is set to the compiled Behavior.

In addition, there may be some generated elements that cannot be stored in conjunction with the context element. In particular, generated Instance Specifications are stored in the nearest Package containing the context element, and generated template instantations are stored in the $$Template Bindings package.

If you have successfully parsed some Alf code text, then you can map that code by calling the *map* method. (If the parse was not successful, then calling *map* has no effect.) Normally, unless there is a system error, mapping should always complete successfully. Since mapping results in an update to the UML model, the *map* method should be called within a MagicDraw session. This can be done conveniently using the *AlfActionUtil.executeSession* method, which also ensures that any automatic Alf compilation is turned off during the session, so new compilations are not accidentally triggered by updates from the mapping process.

```
AlfCompiler compiler = new AlfCompiler();
compiler.setContextElement(element);
compiler.parse(text);
if (compiler.isSuccessful()) {
        AlfActionUtil.executeSession("Map Alf", new Runnable() {
                public void run() {
                        compiler.map();
                        AlfActionUtil.registerDependencies(compiler);
                 }
         });
}
```

Rather than separately parsing and mapping some Alf text, you can attempt to do both with one call to the *compile* method. This method first parses some text for a given context element and, if that is successful, maps it and updates the UML model appropriately with the results of the mapping. If the parse is not successful, then compilation errors are available using the *getCompilerErrors* method, as after a call to the *parse* method. As for the *map* method, the *compile* method should be called within a MagicDraw session, since it may potentially update the UML model if a mapping is carried out.

```
Compiler compiler = new AlfCompiler();
AlfActionUtil.executeSession("Compile Alf", new Runnable() {
        public void run() {
                compiler.compile(element, text);
                AlfActionUtil.registerDependencies(compiler);
        }
});
```

Related Pages

- Compiler API: Parsing