Abstract

The use of *markup languages* is recurrent in the world of technology, with *HyperText Markup Language* (HTML) being the most prominent one due to its use in the Web. The requirement of tools that can automatically build well formed documents with good performance is clear. Yet, the most used solution is *template engines*, which neither ensures well-formed documents nor presents good performance, due to the use of external text files.

To tackle the first issue we propose to define HTML templates as first-class functions instead of using text files. To that end, these HTML template functions use a Java *Domain Specific Language* (DSL) to write HTML. Our main goal is to create the required tools that automatically generate that DSL based on its language definition from an *eXtensive Markup Language Schema Definition* (XSD) file. The resulting DSL should enforce the restrictions of the given language which are specified in the XSD file. By removing the use of text files we are also suppressing the file load overhead and reducing `String` manipulation, which in turn increases the overall performance and solves the second issue.

My proposal, named `xmlet`, includes a set of tools that are able to: 1) parse and extract the rules from a XSD file, 2) generate the adequate classes and methods to define the DSL that reflects the language rules, 3) handle the use of the resulting DSL through the implementation of the Visitor pattern. Finally we validated this solution not only for HTML, but also with the Android *eXtensive Markup Language* (XML) for Android layouts and the regular expressions language.

By comparing the developed solution to some state-of-art solutions, including *template engines* and some other solutions with specific innovations, we obtained

very favorable results with the suggested solution being the best performance-wise in all the tests we performed. These results are important, specially considering that apart from being a more efficient solution it also introduces validations of the language usage based on its syntax definition.

Keywords: XML, eXtensive Markup Language, XSD, eXtensive Markup Language Schema Definition, Automatic Code Generation, Fluent Interface, Domain Specific Language.