SugarJ: Library-based Syntactic Language Extensibility

Sebastian Erdweg, Tillmann Rendel, Christian Kästner, and Klaus Ostermann
University of Marburg, Germany

Abstract
Existing approaches to extend a programming language with syntactic sugar often leave a bitter taste, because they cannot be used with the same ease as the main extension mechanism of the programming language—libraries. Sugar libraries are a novel approach for syntactically extending a programming language within the language. A sugar library is like an ordinary library, but can, in addition, export syntactic sugar for using the library. Sugar libraries maintain the composable and scoping properties of ordinary libraries and are hence particularly well-suited for embedding a multitude of domain-specific languages into a host language. They also inherit self-applicability from libraries, which means that sugar libraries can provide syntactic extensions for the definition of other sugar libraries.

To demonstrate the expressiveness and applicability of sugar libraries, we have developed SugarJ, a language on top of Java, SDF and Stratego, which supports syntactic extensibility. SugarJ employs a novel incremental parsing technique, which allows changing the syntax within a source file. We demonstrate SugarJ by five language extensions, including embeddings of XML and closures in Java, all available as sugar libraries. We illustrate the utility of self-applicability by embedding XML Schema, a metalanguage to define XML languages.

Categories and Subject Descriptors  D.3.2 [Language Classifications]: Extensible languages; D.2.13 [Reusable Software]

General Terms  Languages

Keywords  SugarJ, language extensibility, syntactic sugar, DSL embedding, language composition, libraries

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

OOPSLA'11, October 22–27, 2011, Portland, Oregon, USA.
Copyright © 2011 ACM 978-1-4503-0940-0/11/10...

```java
import pair.Sugar;
public class Test {
    private (String, Integer) p = ("12", 34);
}
```

Figure 1. Using a sugar library for pairs.

1. Introduction
Bridging the gap between domain concepts and the implementation of these concepts in a programming language is one of the “holy grails” of software development. Domain-specific languages (DSLs), such as regular expressions for the domain of text recognition or Java Server Pages for the domain of dynamic web pages have often been proposed to address this problem [31]. To use DSLs in large software systems that touch multiple domains, developers have to be able to compose multiple domain-specific languages and embed them into a common host language [24]. In this context, we consider the long-standing problem of domain-specific syntax [5, 9, 29, 37, 53].

Our novel contribution in this domain is the notion of sugar libraries, a technique to syntactically extend a programming language in the form of libraries. In addition to the semantic artifacts conventionally exported by a library, such as classes and methods, sugar libraries export also syntactic sugar that provides a user-defined syntax for using the semantic artifacts exported by the library. Each piece of syntactic sugar defines some extended syntax and a transformation—called desugaring—of the extended syntax into the syntax of the host language. Sugar libraries enjoy the same flexibility as conventional libraries: (i) They can be used where needed by importing the syntactic sugar as exemplified in Figure 1 (ii) The syntax of multiple DSLs can be composed by importing all corresponding sugar libraries. Their composition may form a new higher-level DSL that can again be packaged as a sugar library. (iii) Sugar libraries are self-applicable: They can import other sugar libraries and the syntax for specifying syntactic sugar can be extended as well.

In other words, sugar libraries treat language extensions in a unified and regular fashion at all metalevels. Here, we apply a conceptual understanding of “metalevel”, which dis-