

## Contents

	Page
<b>Editorial</b>	3
<hr/>	
<b>ARTICLES</b>	
<hr/>	
<b>A Static Analysis to Detect Re-Entrancy in Object Oriented Programs</b> <i>By Manuel Fähndrich, Diego Garbervetsky and Wolfram Schulte</i>	5
<p>The authors present a novel whole program analysis to identify call sites in which a method call is made on an object whose invariants may not hold because of re-entrancy. The analysis augments a points-to analysis to compute potential call chains in order to detect re-entrant calls.</p>	
<b>Adding Type Constructor Parameterization to Java</b> <i>By Vincent Cremet and Philippe Altherr</i>	25
<p>The papers presents a generalization of Java's parametric polymorphism that enables parameterization of classes and methods by type constructors, i.e., functions from types to types. Our extension is formalized as a calculus called <math>FGJ_{\{\omega\}}</math>. It is implemented in a prototype compiler and its type system is proven safe and decidable.</p>	
<b>The Essence of Lightweight Family Polymorphism</b> <i>By Chieri Saito and Atsushi Igarashi</i>	67
<p>Lightweight family polymorphism is a programming style that supports inheritance of mutually recursive classes. The paper gives a translation from .FJ (the core calculus for lightweight family polymorphism) into an extension of Featherweight GJ (the core calculus of generic types) with <i>self type variables</i>. This translation demonstrates the self-type variables are the essence of lightweight family polymorphism.</p>	
<hr/>	
<b>OUTLOOK</b>	
<hr/>	
<b>A brief outlook to the next issue</b>	101