Abordaje funcional a EDSLs
Alberto Pardo y Marcos Viera. Universidad de la República. Uruguay.
Curso en español.

Breve resumen de la materia

Los lenguajes de programación de propósito general no siempre son adecuados para resolver problemas complejos: las soluciones pueden no ser claras y pueden requerir conocimientos avanzados de programación. Una forma de resolver estos inconvenientes es a través de la definición de un lenguaje de dominio específico (DSL: Domain Specific Language), que es un lenguaje hecho a la medida del dominio en que el problema se encuentra. Para evitar el trabajo de diseñar e implementar un lenguaje completamente nuevo para cada dominio, el DSL se puede implementar como una biblioteca del lenguaje "anfitrión". Los lenguajes de esta clase son llamados lenguajes de dominio específico embebidos (EDSL: Embedded Domain Specific Languages). En este curso estudiaremos algunas técnicas de programación funcional que son muy útiles para el diseño e implementación de EDSLs. Este tipo de lenguajes son muy apropiados para la implementación de EDSLs, debido a la existencia de características tales como los tipos de datos algebraicos, funciones de alto orden y sofisticados sistemas de tipos.

Objetivos del curso

El objetivo de este curso es introducir a los estudiantes a los conceptos de DSL y EDSL, y presentar una serie de técnicas para el diseño e implementación de este tipo de lenguajes desde una perspectiva de programación funcional.

Programa

- Lenguajes de Dominio Específico (DSL).
- Lenguajes de Dominio Específico Embebidos (EDSL).
- Shallow embedding y deep embedding.
- Ejemplo de shallow embedding: parsers funcionales.
- Funtores aplicativos.
- Mónadas.
- Ejemplo de deep embedding: lenguaje de expresiones.

Prerrequisitos

Conocimientos básicos de programación funcional (preferiblemente en Haskell).

Bibliografía

- Gibbons, J. (2015). Functional Programming for Domain-Specific Languages. In: Zsók, V., Horváth, Z., Csató, L. (eds) Central European Functional Programming School. CEFP 2013. Lecture Notes in Computer Science(), vol 8606. Springer, Cham. https://doi.org/10.1007/978-3-319-15940-9 1
- McBride, Conor and Paterson, Ross (2008). Applicative programming with effects. J. Funct. Program. 18, 1 (January 2008), 1–13.
- Wadler, P. (1995). Monads for Functional Programming. Advanced Functional Programming (p./pp. 24–52), London: Springer. ISBN: 3-540- 59451-5
- Graham Hutton and Erik Meijer. 1998. Monadic parsing in Haskell. J. Funct. Program. 8, 4 (July 1998), 437–444. https://doi.org/10.1017/S0956796898003050
- García-Garland, J., Pardo, A., Viera, M. (2019). Attribute grammars fly first-class... safer!: dealing with DSL errors in type-level programming. In Stutterheim, J. and Chin, W., editors, IFL '19: Implementation and Application of Functional Languages, Singapore, September 25-27, 2019, pages 10:1–10:12. ACM.
- Kiselyov, O. (2010). Typed Tagless Final Interpreters. SSGIP 2010: 130-174
- Huttner, L., Merigoux, D. (2022) "Catala: Moving Towards the Future of Legal Expert Systems", Artif. Intell. Law.