Relatório - Terceiro Trabalho de Linguagens de Programação

UFJF - 2022

André Caetano Vidal - 201665010AC
Bernardo Souza Abreu Cruz - 201635019

Introdução

Este relatório tem como objetivo descrever e expandir em relação ao terceiro trabalho de Linguagens de Programação no período 2021.3 da UFJF, implementando a linguagem CLASSES usando Racket.
Programas criados utilizando essa linguagem devem consistir de uma sequência de declarações de classe e em seguida uma expressão que poderá fazer uso das classes declaradas. Declarações de classe devem conter nome da superclasse, zero ou mais declarações de campos e zero ou mais definições de métodos. Já a declaração de métodos deve conter nome, lista de parâmetros formais e um corpo.
Além disso, a linguagem deve apresentar quatro expressões para manipulações de instâncias:

- New
- Self
- Send
- Super

Estrutura do Projeto

Durante a realização desse trabalho foi utilizado o arquivo “irefs.rkt” assim como o capítulo 9 do livro “Essentials of Programming Languages”, enviado como material de leitura na disciplina.
Para estruturar o projeto, organizamos a solução em três arquivos principais: classes, irefs e main. Para garantir a execução de testes, utilizamos o arquivo examples para listar os itens que gostaríamos de reproduzir na main. Abaixo vamos passar pelos arquivos “classes.rkt” e “main.rkt”, explicitando informações e o conteúdo criado.

Classes

Em Classes, são criadas as estruturas class, method e object para construir a base da linguagem. As estruturas são compostas da seguinte forma:

- Class: nome da superclasse, nome dos campos e métodos
- Method: argumentos do método, seu corpo, superclasse que está incorporada e seus campos
- Object: nome da classe do objeto e seus campos

Objetos são instanciados a partir do método new-object, trazendo a lista de campos a partir de novas referências. Já para métodos, utilizamos apply-method, relacionando os métodos aqui criados junto a seus parâmetros, com a super classe já existente.
Já as classes são criadas dentro de um ambiente específico, the-class-env, composto por uma lista de listas, em que realizamos operações a partir dos métodos add-to-class-env e lookup-class. O primeiro desses é utilizado dentro da declaração da classe, enquanto ela é criada, adicionando a classe ao ambiente. Já lookup-class é utilizada para buscar e encontrar essa classe, utilizada em diversos outros métodos para garantir que a classe procurada existe e retorná-la para uso imediato.
Além disso, temos dois métodos de inicialização - initialize-class-env! e initialize-class-decl! - criados para inicializar o ambiente e as classes após

declaração, utilizando dos métodos citados anteriormente para garantir o uso das instâncias criadas no programa.
Ainda nesse arquivo, criamos as operações no ambiente de métodos como find-method e merge-methods-env, permitindo que os métodos da classe e da superclasse sejam combinados.
Por último neste tópico, utilizamos alguns métodos auxiliares para determinar o valor de expressões utilizadas na linguagem, estender ambientes e facilmente acessar superclasses a partir de outros métodos.

Main

Expressões

No arquivo main, definimos a especificação do comportamento das expressões usadas, a maioria sendo trazida do “irefs.rkt” como dito anteriormente. Aqui, no entanto, trazemos também a definição das quatro expressões solicitadas no escopo do trabalho, new, self, send e super. As quatro expressões foram definidas da seguinte forma:

- New: Utilizamos para criar um objeto novo de uma classe específica. Aproveitado nas criações de instâncias principalmente no arquivo classes.
- Self: Retorna o objeto aplicado no método atual.
- Send: A partir do objeto passado, procura o método e operação nas classe e superclasse relacionados. Logo em seguida, caso o método seja encontrado, ele é retornado e aplicado em cima do objeto atual.
- Super: Permite encontrar e executar algum método da superclasse a partir do objeto atual.

Execução do Projeto

O projeto gira em torno do método **value-of-program**, separando símbolos relacionados a declaração de classes e da expressão utilizada no programa criado. A partir da leitura e utilizando o método initialize-class-env!, citado

previamente, são armazenados e criados nome da super-classe, campos e métodos relacionados a cada uma das classes declaradas, seguidas da expressões ou operações que gostaríamos de executar no programa com linguagem Classes.

Conclusão

Com base no material estudado, é permitida a execução do programa na linguagem Classes. Para isso, basta declarar as classes desejadas no ambiente, assim como a expressão que deseja operar. Isso deve ser feito dentro do método value-of-program, como mostrado no exemplo criado nos arquivos do trabalho, executando o arquivo “main.rkt”.