The Tyr Programming Language

Timm Felden

August 17, 2017

Abstract

This document defines Tyr, a research language for type-oriented programming. Type-oriented programming is a paradigm that extends on object-oriented programming. In type-oriented languages, types are first order values like integers and objects. An existing but primitive form of type orientation is the Java reflection API.

Acknowledgements

For Pony!

Contents

Part I
Core Language

1 Introduction

Type-oriented programming (TOP) is a paradigm that states that types are objects. In consequence, it is possible to perform calculations on types like any other calculation. As it is true for objects in object-oriented programming (OOP), types can be copied and may have mutable state. The mutable state of a type can be bounded by static knowledge in the same way as pointer can be restricted to point to objects of a certain type. As such, TOP implies OOP.

Tyr is a programming language created to explore this idea in practice. Tyr as a language is a descendant of C++ and Scala. In order to examine consequences of TOP for resource management, Tyr features manual memory management.

2 Syntax

The syntax of Tyr is inspired by Scala.

3 Semantics

The semantics of Tyr is loosely based on C++ and Scala.

var/val: fields type var -> type field (in vtable)

defs: def -> virtual static def -> static type (ada non-poly pointer) type def -> type method

Typen: Any (top) void (<: Any) bool Integer int byte long UnsignedInteger FloatingPoint float double pointer

class Object <: pointer String <: Object IterableOnce <: String Iterable <: IterableOnce Option <: Iterable Seq <: Iterable Array <: Iterable

Part II
Compilation

modules, source paths, modules scopes, default scopes,

module naming convention: <organization>.<project> tyr.lang tyr.system tyr.collection skill.common

Part III
Collections

IterableOnce(T : Type) - static def for (p, b) - def foreach (f : LocalLambda[-> T])

Iterator <: IterableOnce - empty() - move() : bool - get() - for (p, b) = if(!empty) do

Part IV
Appendix