– Добавлена экспериментальная поддержка модели акторов Ractor. Ractor позволяет обеспечить параллельное выполнение программы без хлопот о потокобезопасности. В Ruby 3.0 можно создать несколько ракторов и запустить их параллельно. Общаться ракторы между собой будут с помощью сообщений.

– Добавлен планировщик fiber-потоков Fiber#scheduler. Волокна являются примитивами для реализации легковесного кооперативного параллелизма в Ruby. В основном они являются средством создания блоков кода, которые могут быть приостановлены и возобновлены, как и потоки. Основное отличие заключается в том, что они никогда не вытесняются, а планирование должно выполняться программистом, а не ВМ. Поддерживаемые методы и классы: Mutex#lock, Mutex#unlock, Mutex#sleep, ConditionVariable#wait, Queue#pop, SizedQueue#push, Thread#join, Kernel#sleep, Process.wait, IO#wait, IO#read, IO#write и др.

– Добавлен инструментарий для аннотации типов RBS. Он поддерживает большинство шаблонов в приложениях, написанных на Ruby. Аннотации RBS делают возможным выполнять статический анализ кода без явного определения типов.

– Добавлен экспериментальный анализатор типов TypeProf. На данном этапе TypeProf читает код, анализирует использование методов и генерирует прототип аннотаций типов в формате RBS.

ЛИТЕРАТУРА

1. BigBinary [Электронный ресурс] - Режим доступа: https://www.bigbinary.com/blog/ruby-3-features (дата обращения 10.04.2022).

2. Ruby Лучший Друг Программиста [Электронный ресурс] - Режим доступа: https://www.ruby-lang.org/ru/news/2020/12/25/ruby-3-0-0-released/ (дата обращения 12.04.2022).

УДК 004.045

Студ. С.Н. Дорохов
Науч. рук. А.Д. Томко
(кафедра информационных систем и технологий, БГТУ)

БЛОКИ КАК ОБЪЕКТЫ В RUBY

Блоки – одна из ключевых концепций в Ruby, без которой практически не обходится ни один кусок кода. Они очень похожи на обычные лямбда-функции, но имеют свои особые черты. Для хорошего понимания языка и происходящего в коде нужно понимать, как они устроены. Здесь мы немного копнем вглубь.

Особенности блоков в Ruby.

– С одной стороны, у блоков есть особый синтаксис создания и передачи в функции как особого параметра. С другой стороны, сам блок – это объект, как и всё остальное в языке. Его можно как создать независимо от функции, так и использовать. За блоки в Ruby отвечает класс Proc.

– С объектом-блоком можно делать всё то же самое, что и с другими объектами. В этом смысле он ведет себя как анонимная функция в любом языке. Однако, если мы захотим этот объект использовать как блок при передаче в функцию, то ничего не получится.

– Хотя мы и имеем дело с блоком, всё же в примере выше он передается в функцию как обычный объект первым параметром. Но метод map() не принимает на вход ничего, кроме блока, поэтому код завершается с ошибкой. Блок, созданный как объект, невозможно напрямую использовать в методах, ожидающих на вход блоки. Для этого нужен специальный синтаксис.

– Амперсанд, добавленный в начале переменной, содержащей блок, передает этот блок в функцию не как параметр, а как блок.

ЛИТЕРАТУРА

1. Ruby Лучший Друг Программиста [Электронный ресурс] - Режим доступа: https://www.ruby-lang.org/ru/news/2020/12/25/ruby-3-0-0-released/ (дата обращения 12.04.2022).

2. Ruby. Объектно-ориентированное проектирование [Электронный ресурс] — Режим доступа: https://codernet.ru/books/ruby/ruby_obektno-orientirovannoe_proektirovanie_sendi_metc/.

УДК 004.932

Студ. А.В. Прудилко
Науч. рук. А.Д. Томко
(кафедра информационных систем и технологий, БГТУ)

ВЕБ-ПРИЛОЖЕНИЕ «ФОТОХОСТИНГ»

В ходе работы была поставлена цель продемонстрировать веб-приложение, созданное в рамках дипломного проектирования.

Веб-приложение «Фотохостинг» представляет собой монолитное веб-приложение, созданное на языке программирования «Python» с использованием веб-фреймворка «Django», в качестве хранилища данных была выбрана одна из востребованных баз данных – «MySQL». Веб-приложение развернуто в Docker-контейнере. Приложение опубликована HTTP-сервер «Gunicorn» в связке с прокси-сервером «Nginx». Архитектура веб-приложения на Django основывается на структурном