Федеральное агентство связи

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования «Московский технический университет связи и информатики»

**Кафедра Математической кибернетики и информационных технологий**

Документация к практической работе

«Создание децентрализованного приложения с применением технологии blockchain на платформе Waves»

Выполнил студент

группы БВТ1702

Жмакин Е.В.

Проверил: Городничев М.Г.

**Оглавление**

[1. Обзор платформы Waves 3](#_Toc13263853)

[2. Язык RIDE 3](#_Toc13263854)

[2.1 Введение 3](#_Toc13263855)

[2.2 Скрипты 4](#_Toc13263856)

[2.3 Переменные и типы данных 5](#_Toc13263857)

[2.4 Функции 6](#_Toc13263858)

# Обзор платформы Waves

**Waves Platform** - открытая блокчейн платформа, разработанная для простоты использования и массового внедрения технологии блокчейн. Это комплексная блокчейн-экосистема, которая предоставляет все необходимые функции для внедрения технологии блокчейн в бизнес.

Платформа построена на базе модульного фреймворка Scorex, разработанного на языке Scala.

Платформа предоставляет возможность создания смарт-контрактов. Смарт-контракты включают в себя контроль над аккаунтом и над токенами, и предоставляют всё необходимое для реализации таких функций, как кошелёк с мультиподписью, atomic swap, двухфакторная аутентификация. Также Waves вводит новый тип транзакции - “Data Transaction”. Data Transaction - это способ публикации Oracle-данных в блокчейн при помощи смарт-контракта. Для создания смарт-контрактов на этой платформе используется язык программирования RIDE.

# Язык RIDE

## Введение

**RIDE** — функциональный язык программирования, основанный на выражениях.

Язык:

* имеет сильную статическую типизацию
* не имеет циклов, рекурсий и goto-подобных выражений, и поэтому является неполным по Тьюрингу
* по замыслу ленивый

## Скрипты

**Скрипт на RIDE** — код на языке RIDE.

Существует три вида скриптов на RIDE:

* **dApp-скрипт** — скрипт на RIDE, можно привязать к аккаунту на платформе с помощью транзакции установки скрипта, создав таким образом dApp-приложение.
* **Скрипт аккаунта** — скрипт на RIDE, прикрепляется к аккаунту с помощью транзакции установки скрипта. Аккаунт с прикрепленным к нему скриптом называется смарт-аккаунтом.
* **Скрипт ассета** — скрипт на RIDE, прикрепляется к ассету с помощью транзакции установки скрипта ассета. Ассет с прикрепленным к нему скриптом называется смарт-ассетом. Прикрепить скрипт к ассету можно только в момент создания ассета.

**Аккаунтом** называется математически связанная пара публичного и приватного ключей. Пара считается аккаунтом, если публичный ключ пары участвовал хотя бы в одной транзакции в блокчейне, т.е. записан хотя бы в один блок.

**Ассет** — объект из реального или виртуального мира, имеющий ценность, обеспеченный чем-либо. В контексте блокчейна Waves ассет является синонимом слова токен — слова ассет и токен используются взаимозаменяемо.

Cложность скрипта на RIDE — безразмерная величина, которая оценивает вычислительные ресурсы необходимые для исполнения скрипта на RIDE. Сложность скрипта на RIDE оценивается исходя из сложностей всех операторов и функций, составляющих скрипт. Сложность скрипта в настоящий момент ограничена 4000.

## Переменные и типы данных

В языке RIDE **все переменные являются неизменяемыми** (immutable variables). Это значит, что как только мы присвоили значение переменной, ее значение в будущем уже нельзя будет изменить. Переменные в RIDE работают, как переменные-константы в основных языках программирования.

**let**— ключевое слово, с помощью которого неизменяемой переменной присваивается значение *ленивым образом* (слово let работает, как lazy val в языке Scala). Значение переменной, определенной с помощью ключевого слова let, будет вычислено только в момент ее первого использования.

Все типы данных в RIDE можно разбить на следующие группы:

1. Базовые типы
2. Специальные типы
3. Предопределенные структуры данных

**Базовые типы**

1. Boolean
2. ByteVector
3. Int
4. String

**Специальные типы**

1. List
2. Nothing
3. Unit

**Предопределенные структуры данных**

1. Address
2. Alias
3. DataEntry
4. ScriptResult
5. ScriptTransfer
6. TransferSet
7. WriteSet

## Функции

Функция должна возвращать значение. В сигнатуре функции тип возращаемого значения не указывается. Функция должна быть объявлена выше места ее использования.

Пример определения функции с двумя параметрами:

func main(amount: Int, name: String) = {

throw()

}