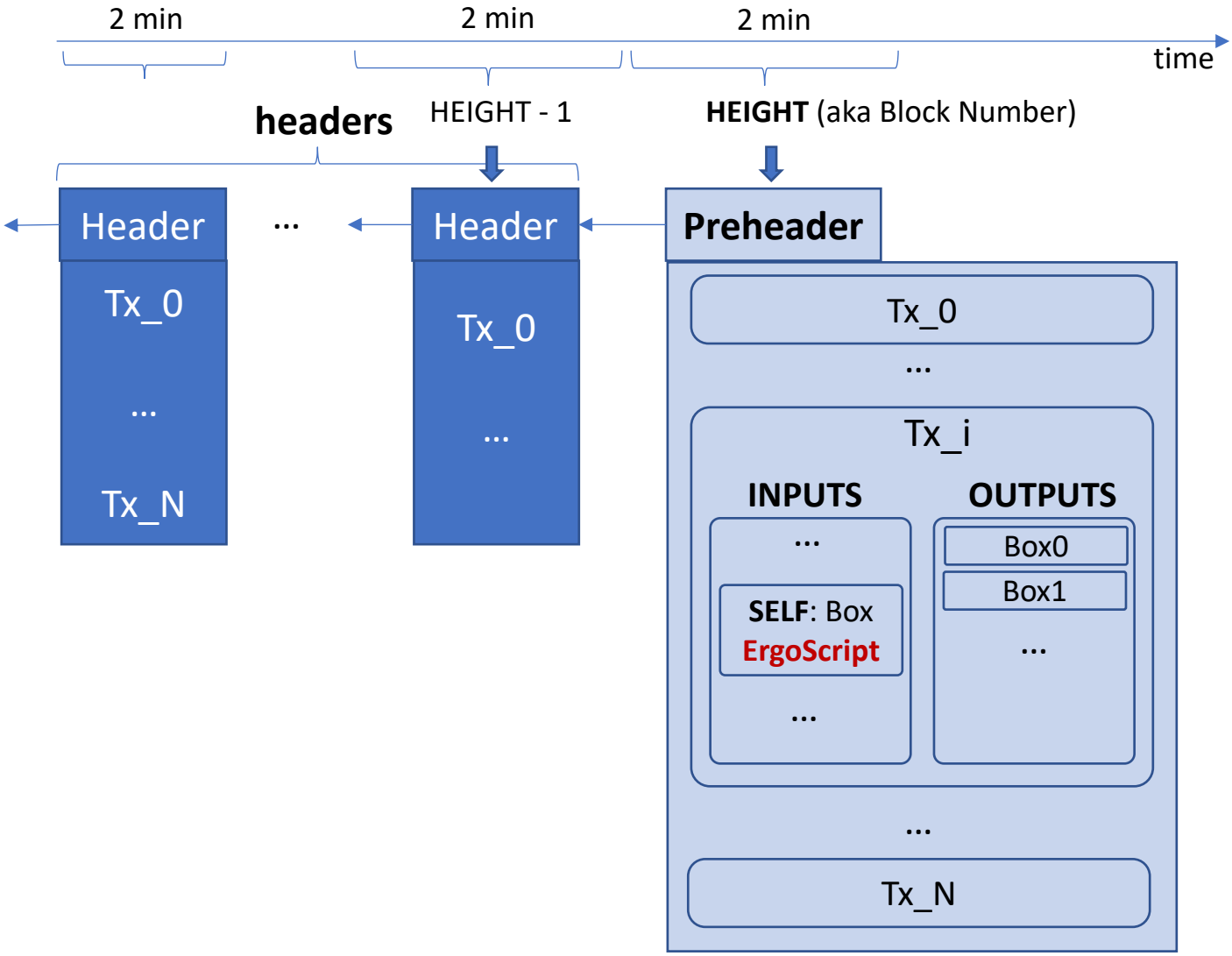
ErgoScript

Александр Слесаренко

@aslesarenko

ErgoScript in Ergo blockchain



ErgoScript Language

literals

```
val ok = true
val count: Int = 1
val sent: Long = 350L
val items = Coll(1,2,3)
val unit: Unit = ()
val tuple = (1L, 2, true)
```

expressions

```
val y = 5 + x * 3
if (check) happy else sad
tuple._1 + f(tuple._2)
coll(i) + coll(j)
val z = { val x = f(1); x + y }
```

functions

```
val f = {(x: Int) => x.toLong}
def swap(x: (Int, Long)) = (x._2, x._1)
```

types

Boolean	logical
Byte, Short, Int, Long, BigInt	numeric
GroupElement	elliptic curve
SigmaProp	Sigma protocol proposition
(T1,…, Tn)	tuples
Coll[T]	collections
Header, Preheader, Context, Box, AvlTree	Pre-defined classes

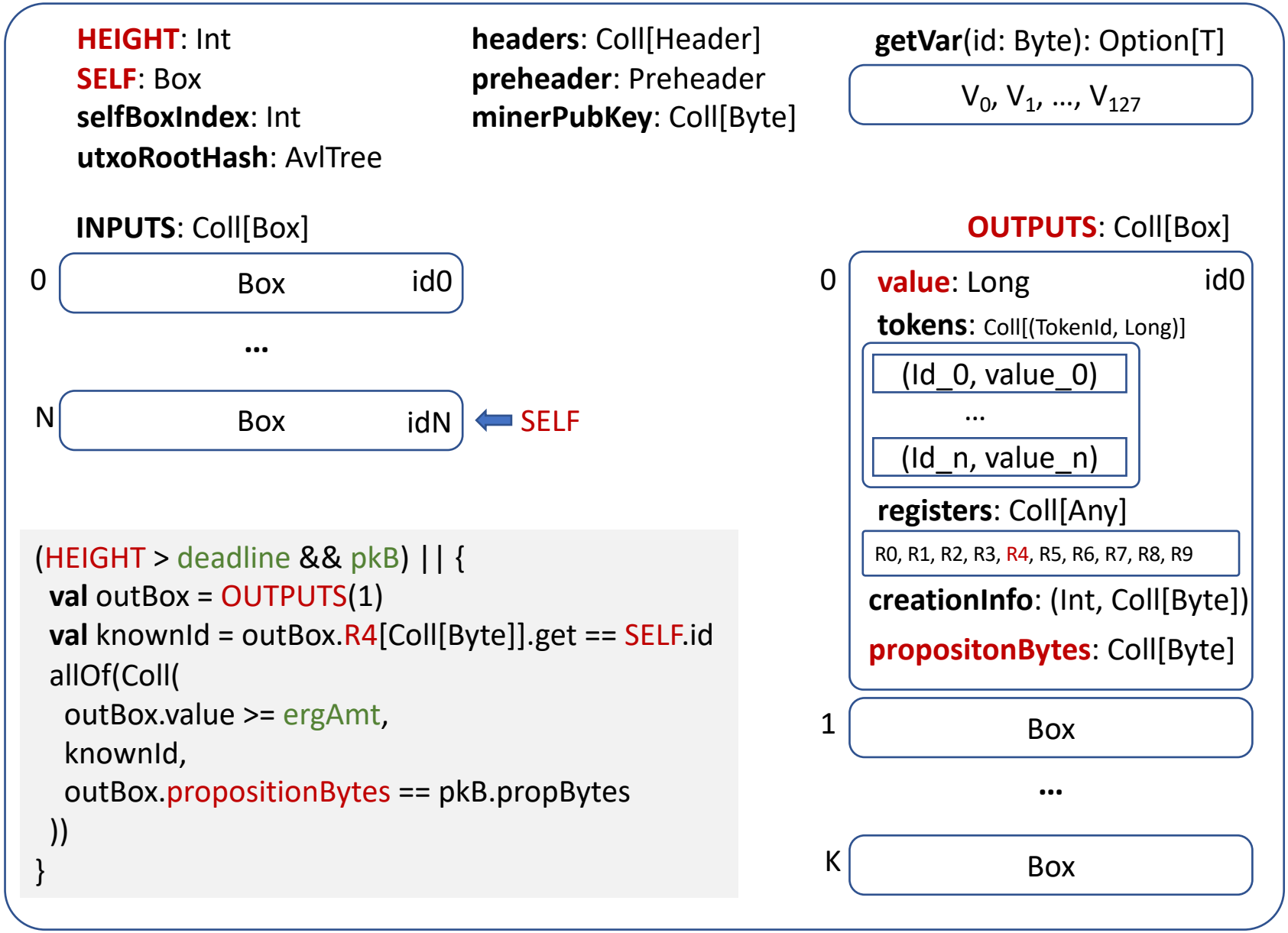
collection operations

```
xs.map {(x: Int) => x + 1}
xs.filter {(x: Int) => x + 1}
xs.foldLeft(0){(acc: Int, x: Int) => acc + x}
val pairs = keys.zip(values)
unzip(pairs)._1 == keys
INPUTS.flatMap(_.tokens).sumByKey.getByKey(tokenId)
```

sigma propositions

```
(pkAlice && HEIGHT < deadline) ||
(pkBob && HEIGHT >= dealine)
```

ErgoScript execution context




```
(HEIGHT > deadline && pkB) || {
  val outBox = OUTPUTS(1)
  val knownId = outBox.R4[Coll[Byte]].get == SELF.id
  allOf(Coll(
    outBox.value >= ergAmt,
    knownId,
    outBox.propositionBytes == pkB.propBytes
  ))
}
```

Example: Sell Tokens for Ergs

Алиса хочет купить токен (tid) в количестве tAmt за ergAmt Ergo.
Для этого она кладет ergAmt в специальный ящик и защищает его таким скриптом:

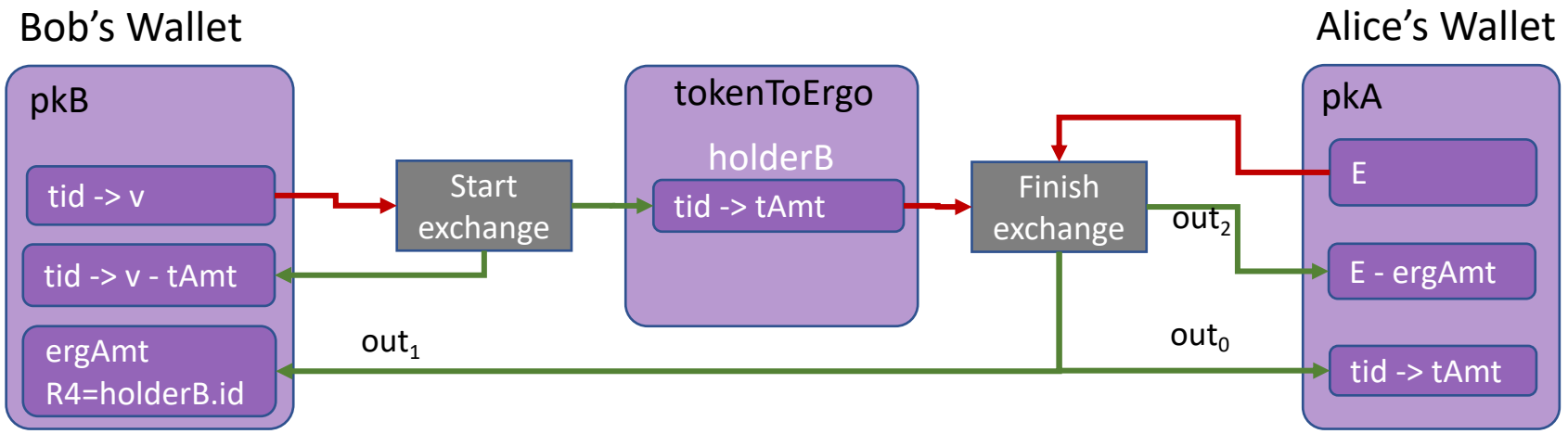
ergoToToken contract

```
(HEIGHT > deadline && pkA) || {
  val outBox = OUTPUTS(0)
  val tokenData = outBox.tokens.byKey(tid)
  val knownId = outBox.R4[Coll[Byte]].get == SELF.id
  allOf(Coll(
    tokenData._2 >= tAmt,
    outBox.propositionBytes == pkA.propBytes,
    knownId
  ))
}
```

Bob хочет продать токен (tid) в количестве tAmt за ergAmt Ergo.
Для этого он кладет (tid, tAmt) в специальный ящик и защищает его таким скриптом:

tokenToErgo contract

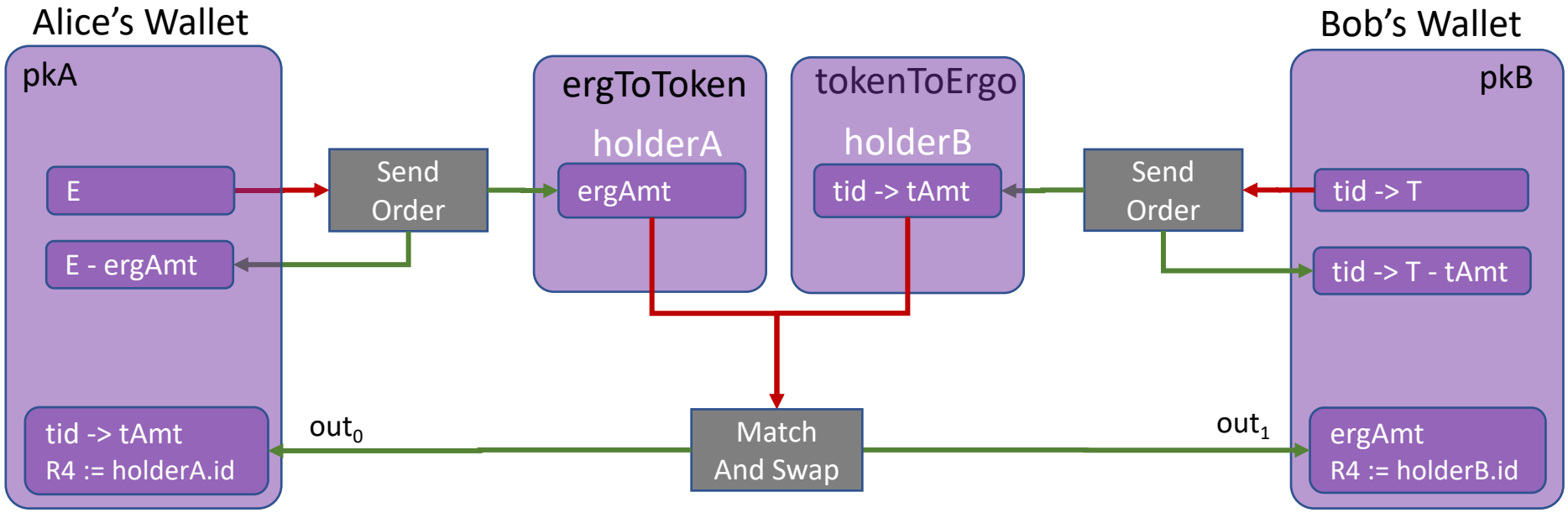
```
(HEIGHT > deadline && pkB) || {
  val outBox = OUTPUTS(1)
  val knownId = outBox.R4[Coll[Byte]].get == SELF.id
  allOf(Coll(
    outBox.value >= ergAmt,
    knownId,
    outBox.propositionBytes == pkB.propBytes
  ))
}
```



Example: DEX matching (simple)

Алиса хочет купить токен (tid) в количестве tAmt за ergAmt Ergo.
Для этого она кладет ergAmt в специальный ящик и защищает его ergoToToken contract (SendOrder transaction).

Bob хочет продать токен (tid) в количестве tAmt за ergAmt Ergo.
Для этого он кладет (tid -> tAmt) в специальный ящик и защищает его tokenToErgo contract (SendOrder transaction)



DEX наблюдает за блокчейном и обнаруживает подходящую пару боксов. DEX формирует транзакцию обмена (MatchAndSwap), которая удовлетворяет сразу обоим контрактам (валидирует соответствующие условия траты), и при этом не требует подписи.
В результате в кошельке Алисы и Боба появятся соответствующие боксы.
Отметим, на Ergo, pkA, pkB могут быть любые sigma proposition