REFERENCE

This reference may serve to give a first-hand impression about how Lexon works concretely. It is still in flux and best looked up online when the intent is to create real code.

http://lexon.tech/reference

• (THE DOT)

end of statements

<statement>.

<expression>.

The dot signifies the end of a statement or expression. It is explained with the respective grammar. It cannot be used as part of definition names or in any other function. The dot is usually also the end of a line.

Dots do not matter in COMMENTS.

GRAMMAR

This is a word that is part of the meta structure of Lexon code. It groups and gives meaning to the terms following it.

EXAMPLE

"Grantor" is a person.

LEXON 0.1

A

optional article to increase readability

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

EXAMPLE

```
"Grantor" is a person.
```

VARIANTS

AN

SYNONYMS

THE

LEXON 0.1

ABORTED

end the performance

PERFORMANCE

This term controls the performance of the code, e.g. if following statements apply, or not; or if the contract should terminate.

LEXON 0.2

EXAMPLE

```
And with this, the
Proposal is aborted.
```

AFTER

later in time

COMPARISON

This is an comparison operator to compare two values. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

LEXON 0.3

EXAMPLE

The Due Date is set as the Delivery Time after the current time.

AFTERWARDS

conditional and chronological order

<statement>, and afterward
<statement>

indicates that the following statement should come into effect if the preceding was performed. If the preceding statement could not be performed, the following statement should not even be looked at.

SEQUENCE

This word puts statements before and after it in chronological order. This is a reserved keyword that has significant meaning to the Lexon transpiler and virtual machine. It cannot be used as a variable or contract name. It does not matter whether any of its letters are written in lower or upper case. Other words can include this keyword as part.

LEXON 0.2

EXAMPLE

```
and afterwards publish
Owner and Cost.
```

VARIANTS

AND AFTERWARDS

SYNONYMS

THEREBY, AND ALSO

ALSO

conditional and chronological order

also <provision>

SEQUENCE

This word puts statements before and after it in chronological order. This is a reserved keyword that has significant meaning to the Lexon transpiler and virtual machine. It cannot be used as a variable or contract name. It does not matter whether any of its letters are written in lower or upper case. Other words can include this keyword as part.

LEXON 0.2

EXAMPLE

```
Also provided that the
Owner is not This
Contract.
```

AMOUNT

a number

"<name>" is [an] amount.

This keyword is used to state that a name will mean a number, e.g. amounts of crypto currency.

TYPE

This is a type name, which is used to clarify what kind of data which name definition might stand for. A type name can be used like a definition, i.e. a name could be defined that has the same name like its type. But no definition may have the name of a type other than its own.

LEXON 0.1

EXAMPLE

```
"Inheritance" is an
amount.
```

AMOUNT OF

optional indicator that the value and not the name is used for a calculation

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.3

EXAMPLE

```
Increase the count of
Votes by the amount of
Nays.
```

AND

conditional and chronological order

<binary> and <binary>

<statement> and <statement>

And is used similar to its meaning in natural language, for concatenation of two or more instructions and also in conjunction with commas. Different to most other program languages, 'and' can allow to re-use a subject.

SEQUENCE

This word puts statements before and after it in chronological order. This is a reserved keyword that has significant meaning to the Lexon transpiler and virtual machine. It cannot be used as a variable or contract name. It does not matter whether any of its letters are written in lower or upper case. Other words can include this keyword as part.

LEXON 0.1

EXAMPLE

```
The Executor may pay
the Escrow to the Heir
and terminate this
contract.
```

AND ALSO

conditional and chronological order

<statement> and also <statement>

indicates that the following statement should come into effect if the preceding was performed. If the preceding statement could not be performed, the following statement should not even be looked at.

Commas can be used to chain multiple statements together that are each understood to be temporally ordered by and also from the first to the last.

SEQUENCE

This word puts statements before and after it in chronological order. This is a reserved keyword that has significant meaning to the Lexon transpiler and virtual machine. It cannot be used as a variable or contract name. It does not matter whether any of its letters are written in lower or upper case. Other words can include this keyword as part.

EXAMPLE

```
The Payer pays an
Amount into escrow,
appoints the Payee,
and also fixes the Fee.
```

SYNONYMS

THEREBY, AFTERWARDS

LEXON 0.2

AND WITH THIS

conditional and chronological order

<statement> and with this, <statement>

SEQUENCE

This word puts statements before and after it in chronological order. This is a reserved keyword that has significant meaning to the Lexon transpiler and virtual machine. It cannot be used as a variable or contract name. It does not matter whether any of its letters are written in lower or upper case. Other words can include this keyword as part.

LEXON 0.2

EXAMPLE

```
And with this, the
Proposal is Aborted.
```

ANY

optional article to increase readability

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.3

EXAMPLE

```
Redefine the Possessor
to be any person.
```

ANYONE

blanket permission

anyone may <statement>

PERMISSION

This keyword allows to articulate who should be allowed to initiate the performance of a given clause.

EXAMPLE

```
Anyone may: if Next Tax
Due Date has passed,
then:
Trigger The Default.
```

LEXON 0.3

APPOINT

assign meaning to a name

<person> appoint[s] <person> to be <person>.

Used to define the meaning of a name. Appoint works only for names that did not have a meaning assigned before. To change a name's meaning, use 'redefine' or 'change'.

Though it would not be sensible natural grammar, appoint can be used to define other than person names.

ASSIGNMENT

This term is used to define the value of a name.

EXAMPLE

The Grantor appoints a Person to be Executor.

VARIANTS

APPOINT AS / APPOINT .. TO BE / APPOINTS .. TO BE

LEXON 0.2

AT ALL TIMES PROVIDED

general verification condition

at all times provided [that] <binary>

VERIFICATION

This term allows for formal verification, enforcing that certain names may have only certain values under certain conditions. Performance of code stops when a verification fails.

LEXON 0.6

EXAMPLE

At all times provided that the Owner is This Contract or the Escrow is greater or equal to the Minimal Cost.

AT ANY TIME

access control

FILLER

This keyword allows to articulate who should be allowed to initiate the performance of a given clause.

LEXON 0.2

EXAMPLE

```
CLAUSE: Change.
At any time the Grantor
may fix the Share.
```

AT LEAST

size of a number

COMPARISON

This is an comparison operator to compare two values. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

LEXON 0.3

EXAMPLE

```
The Initiator sets the
"Processing Reward" to
any amount that is at
least 0.
```

BE

type definition, assignments and comparisons

"<name>" be [a] <type>.
"<name>" be <value>.

1. Used to signify that a name will have the meaning of a given type, like a time (type: time) or a number (type: amount).
2. Part of multiple grammar constructs to define or change the meaning of names.

ASSIGNMENT

This term is used to define the value of a name.

SYNONYMS

IS

EXAMPLE

```
"Grantor" be a person.
"Due Date" be
tomorrow.
```

LEXON 0.2

BE MADE

optional part of a predicate to increase readability

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.6

EXAMPLE

```
A Status Notification be made.
```

BEFORE OR ON

earlier in time

COMPARISON

This is an comparison operator to compare two values. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

LEXON 0.3

EXAMPLE

The Service Provider may, if Time of Provision of Services is before or on Due Date, and Provision of Services Have Met the Defined Service Criteria then pay the Service Fee from escrow to themselves, and also pay the Fee from escrow to the Assessor.

BEING

same value

COMPARISON

This is an comparison operator to compare two values. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

LEXON 0.3

EXAMPLE

CLAUSE: Voting Phase Expired.
"Voting Phase Expired" is defined as the Current Period Number being greater than the Last Voting Phase Period Number.

BEING ON RECORD

optional term to increase readability

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.3

EXAMPLE

"Having Never Voted Yes" is defined as no Latest Yes Vote being on record.

BINARY

yes or now / true or false

"<name>" is binary.

TYPE

This is a type name, which is used to clarify what kind of data which name definition might stand for. A type name can be used like a definition, i.e. a name could be defined that has the same name like its type. But no definition may have the name of a type other than its own.

LEXON 0.3

EXAMPLE

"Exceptional Status" is binary.

BURN

delete tokens

burn [the] escrow
burn <number> of <name>

CRYPTO

This term articulates the sending and receipt of crypto currency and any type of token.

LEXON 0.3

EXAMPLE

```
Burn the remainder of
the escrow.
```

CALCULATE

execute and output result

calculate <formula>

OUTPUT

This is a keyword that signals a result that the performance of the code should share with the world: e.g. print to the screen or write to the blockchain receipt log. Results can often be encrypted to facilitate controlled sharing of private information.

LEXON 0.3

EXAMPLE

```
CLAUSE: Current Period Number.
The "Current Period Number" is defined as the whole number resulting from calculating the time passed since Summoning Time, divided by the Period Duration.
```

CERTIFY

optional predicate

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

EXAMPLE

A Member may certify a Vote.

VARIANTS

BE CERTIFIED

SYNONYMS

SIGN

LEXON 0.2

CLAUSE

start of a clause of the code

CLAUSE: <name>.

CLAUSE starts a subsection of code or an algorithmic definition. It is followed by a name of the subsection or definition. This name can be used to allow state transition of the contract system: in terms of the blockchain implementation of the contract, that a person triggers this clause from a browser interface and with this, initiates the changes to the contact state that the clause might describe. Not all clauses are for this purpose, others are simply groups of statements that are relevant for other clauses or used to facilitate nesting of decisions trees. For example, Lexon really allows for only one if-statement per clause. To nest multiple if-statements, the lower order ones will be subsurmized under their own clause.Good style will use clauses to replace definitons that include calculations.

From the perspective of programming, the CLAUSE keyword serves as a 'function head' and to separate the logic of a script into smaller units, in a fine granulity as used in functional languages.

GRAMMAR

This is a word that is part of the meta structure of Lexon code. It groups and gives meaning to the terms following it.

LEXON 0.2

EXAMPLE

```
CLAUSE: Set Up.
```

SYNONYMS

CHAPTER

COLLECT

withdraw tokens

collect <amount> from <person>

CRYPTO

This term articulates the sending and receipt of crypto currency and any type of token.

LEXON 0.3

EXAMPLE

Collect the Proposal Deposit in Approved Tokens from the Proposer to the escrow.

COMMENT

non-processing text

COMMENT: <comment text>

Anything after the COMMENT keyword is ignored for the contract performance until the next line break. Dots are assumed to be part of a comment. The comment can be multiple lines on a printout or on-screen if the program editor or the word editor chose to display text wrapped across multiple lines. The comment still ends only after the place where the enter key had been hit to start a new line.

GRAMMAR

This is a word that is part of the meta structure of Lexon code. It groups and gives meaning to the terms following it.

LEXON 0.1

EXAMPLE

```
COMMENT: A last will
with multiple recipients
```

CONSIDER

set binary true

<name> is considered <state>.

ASSIGNMENT

This term is used to define the value of a name.

LEXON 0.3

EXAMPLE

The Proposal is considered Processed.

CONTRACTS

code relating to individual legal agreements

CONTRACTS [as] per <name>:

All code after the CONTRACTS keyword is interpreted to define the 1:1 relationship between two contracting partys. Code part for the 1:1 logic between contract partners, as opposed to the TERMS part that holds statements that are relevant across all individual contracts.

This is mainly an issue of perspective. Definitions under CONTRACTS are per-agreement: they usually exist multiple times eventually, once for every contract.

GRAMMAR

This is a word that is part of the meta structure of Lexon code. It groups and gives meaning to the terms following it.

EXAMPLE

```
CONTRACTS as per
Partner:
```

LEXON 0.2

COUNT OF

optional indicator that the value and not the name is used for a calculation

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.3

EXAMPLE

```
Increase the count of No Votes by the number of the Shares of the Member.
```

CURRENT

optional indicator that a value as recorded at the time is referred to

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.6

EXAMPLE

```
The "Current Tax" be
defined as the product
of the current Cost and
the current Tax.
```

CURRENT TIME

point in time

current time

VALUE

This is a value that a name can be defined to have or that the meaning of a name can be compared against. It can also be used to articulate verifications.

LEXON 0.2

EXAMPLE

Data be certified,
with Time fixed as the
current time.

DATA

a hash

"<name>" is data

This keyword is used to state that a name stands for the unique identifier for a data set, e.g. its cryptographic hash.

TYPE

This is a type name, which is used to clarify what kind of data which name definition might stand for. A type name can be used like a definition, i.e. a name could be defined that has the same name like its type. But no definition may have the name of a type other than its own.

EXAMPLE

```
"Data" is data.
```

VARIANTS

DATA FINGERPRINT

SYNONYMS

CRYPTOLOGICAL
FINGERPRINT

LEXON 0.2

DATE

a point in time, expressed as a calendar day

"<name>" is [a] date

This keyword is used to state that a name will mean a time, consisting of a date and a time with precision of one second.

TYPE

This is a type name, which is used to clarify what kind of data which name definition might stand for. A type name can be used like a definition, i.e. a name could be defined that has the same name like its type. But no definition may have the name of a type other than its own.

EXAMPLE

```
"Deadline" is a date.
```

LEXON 0.1

DAY

time duration

day

UNIT

This is a unit for values, it can also be used to mean 1 unit of this.

LEXON 0.3

EXAMPLE

"Tax Frequency" is a time duration, greater than 1 day.

DECREASE

subtraction

decrease <name> by <value>
<name> are decreased by <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.2

EXAMPLE

Decreased Owned Shares by the Given Amount.

DECREASED BY

subtraction

decrease <name> by <value>
<name> are decreased by <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.2

EXAMPLE

The "Last Voting Phase Period Number" be defined as the sum of the Starting Period Number and the Voting Phase Duration In Periods decreased by 1.

DEEM

set binary true

<name> is deemed <state>

ASSIGNMENT

This term is used to define the value of a name.

LEXON 0.2

EXAMPLE

The Proposal is deemed Passed.

DEFINE

assign meaning to a name

"<name>" is defined to be <value>.

Used to define the meaning of a name.

Define works only for names that did not have a meaning assigned before. To change a name's meaning, use 'redefine' or 'change'.

ASSIGNMENT

This term is used to define the value of a name.

VARIANTS

DEFINES / DEFINED / DEFINED TO BE

SYNONYMS

APPOINT, FIX

LEXON 0.2

EXAMPLE

"Due Date" is defined to be Tomorrow.

DIFFERENCE

result of subtraction

difference between <value> and <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.2

EXAMPLE

The Maintainer may: pay the difference between the amount in Escrow and the Minimal Cost from the Escrow to themselves.

DIVIDED BY

mathematical division

<value> devided by <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.2

EXAMPLE

```
t0 plus t2 to the power of
2 divided by t2.
```

DIVIDING

mathematical division

dividing <value> by <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.3

EXAMPLE

The "Voting Phase Periods" is defined as the whole number resulting from dividing the Voting Phase Duration by the Period Duration.

DURATION

a duration of time, precise to the second

"<name>" is [a] duration of time.

This keyword is used to state that a name will mean a duration of time, expressed in years, months, weeks, days, hours, minutes or seconds.

TYPE

This is a type name, which is used to clarify what kind of data which name definition might stand for. A type name can be used like a definition, i.e. a name could be defined that has the same name like its type. But no definition may have the name of a type other than its own.

LEXON 0.2

EXAMPLE

"Period" is defined as a duration of a fifth of a day.

ENTER

create a new legal contract

PERFORMANCE

This term controls the performance of the code, e.g. if following statements apply, or not; or if the contract should terminate.

LEXON 0.3

EXAMPLE

```
The Summoner enters
into a Member Contract
with Summoner's Initial
Number Of Shares.
```

ESCROW

the balance of a smart contract, over all individual individual contracts

"<name>" is [an] escrow

This keyword is used to state that a variable contains the special value of the crypto currency balance that is held in the contract.

TYPE

This is a type name, which is used to clarify what kind of data which name definition might stand for. A type name can be used like a definition, i.e. a name could be defined that has the same name like its type. But no definition may have the name of a type other than its own.

EXAMPLE

```
"Inheritance" is an
escrow.
```

VARIANTS

INTO ESCROW / FROM ESCROW

LEXON 0.2

FIFTH

result of division by five

fifth [of] [a] <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.3

EXAMPLE

"Period Duration" is defined as a duration of a fifth of a day.

FIX

assign meaning to a name

<person> fix[es] <name>

ASSIGNMENT

This term is used to define the value of a name.

SYNONYMS

SET, APPOINT

LEXON 0.2

EXAMPLE

The Provider fixes the Fee.

FOR ALL

operation across all contracts

for all <name> <clause>

AGGREGATION

This is an aggregation of all values across all contracts of a given type, e.g. summing up individual balances.

LEXON 0.2

EXAMPLE

```
CLAUSE: Execute.
The Executor may
for all Heirs Payout.
```

FOURTH

result of division by four

fourth [of] [a] <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.3

EXAMPLE

"Period Duration" is defined as a duration of a fourth of a day.

FURTHER IN THE FUTURE THAN

later in time

further in the future than <duration of time>

COMPARISON

This is an comparison operator to compare two values. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

LEXON 0.6

EXAMPLE

```
At all times provided that
Next Tax Due Date is not
further in the future than
the number of days
given as Tax Frequency.
```

GIVEN

optional indicator that a value as recorded at the time is referred to

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.6

EXAMPLE

```
At all times provided that
Next Tax Due Date is not
further in the future than
the number of days
given as Tax Frequency.
```

GREATER OR EQUAL TO

compare two values

greater or equal [to] <value>

COMPARISON

This is an comparison operator to compare two values. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

LEXON 0.1

EXAMPLE

"Cost" is an amount,
greater or equal to 0.

GREATER THAN

compare two values

greater than <value>

COMPARISON

This is an comparison operator to compare two values. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

LEXON 0.1

EXAMPLE

"Cost" is an amount, greater than 0.

HALF

result of division by two

half [of] [a] <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.3

EXAMPLE

"Period Duration" is defined as a duration of a half day.

HAVING BEEN

binary being true

having been <binary>

COMPARISON

This is an comparison operator to compare two values. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

LEXON 0.3

EXAMPLE

```
Eligibility To Ragequit is
defined as Having Never
Voted Yes
or the Proposal of the
Latest Yes Vote having
been Processed.
```

IF ... THEN: ... ELSE:

optional code branches

PERFORMANCE

This term controls the performance of the code, e.g. if following statements apply, or not; or if the contract should terminate.

LEXON 0.2

EXAMPLE

```
If the Possessor is not the
Owner,
then: Hand Over to the
Exception Handler;
else: Set Next Tax Date,
afterwards Update
Records,
and afterwards publish
Owner and Cost.
```

IN

diverse uses as filler and part of other key word terms

MULTIPLE MEANINGS

The word "in" is part of other keywords / key terms.

LEXON 0.2

EXAMPLE

If in Exceptional Status, the Exception Handler may:
redefine the Possessor to be any person.

An Applicant may offer a Token Tribute in Approved Tokens, set the Shares Requested, set the Details, and by this Create a Proposal with the Shares Requested.

IN ANY CASE

optional reaffirmation that all sentences in a clause are independent declarations

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.2

EXAMPLE

```
In any case, afterwards
terminate this contract.
```

INCREASE

mathematical addition

increase <name> by <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

EXAMPLE

The Total Shares Requested are increased by the Shares Requested.

increase the count of No Votes by the number of the Shares of the Member.

LEXON 0.2

INCREASED BY

mathematical addition

<value> increased by <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.2

EXAMPLE

The Starting Period Number be defined as the Last Blocked Period Number increased by 1.

INVOKE

set binary true

<binary> be invoked.

ASSIGNMENT

This term is used to define the value of a name.

VARIANTS

BE … INVOKED

LEXON 0.3

EXAMPLE

```
Exceptional Status be
invoked.
```

IS

type definitions, assignments and comparisons

"<name>" is [a] <type>.
If <definition> is <value>, then:
<statements>.
"<name>" is redefined to be <value>.

1. Used to signify that a variable will have the content of a given type, like a time (type: time) or a number (type: amount).
2. Part of multiple grammar constructs to redefine definitions or make comparisons.
3. Test of existance in a specific list, i.e. having certain attributes.

ASSIGNMENT / COMPARISON

This is an comparison operator to compare two values. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims. / This term is used to define the value of a name.

EXAMPLE

"Due Date" is redefined to be tomorrow.

If the Applicant is not a Member then, Enlist Applicant as Member.

LEXON 0.2

LESS OR EQUAL TO

compare two values

less or equal [to] <value>

COMPARISON

This is an comparison operator to compare two values. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

LEXON 0.1

EXAMPLE

"Cost" is an amount, less or equal to 100.

LESS THAN

compare two values

less than <duration of time>

COMPARISON

This is an comparison operator to compare two values. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

LEXON 0.1

EXAMPLE

"Cost" is an amount, less than 100.

LEX

start of lexon code

LEX <name>.

LEX is the keyword that starts the Lexon script. It is mandatory and it is immediately followed by the name for the smart contract (or contract system consisting of multiple contracts) that is being defined. After the name, a dot is expected. Definitions, and CLAUSES will usually follow below, often divided in a TERMS and CONTRACTS section.

GRAMMAR

This is a word that is part of the meta structure of Lexon code. It groups and gives meaning to the terms following it.

LEXON 0.2

EXAMPLE

```
LEX Will.
```

LEXON

script version follows

LEXON: <version>

This optional tag helps to stay oriented about the different versions of Lexon that code and code examples are made for. E.g. an example for Lexon 0.2 might not work for 0.3 and actually be misleading.
This is aspect is important enough in practice that it warrants its own keyword. In general, older code will often not run with newer versions of the compiler, as is normal for programming languages. The expectation is that code within one 'minor' version number (e.g. the 2 in 0.2.0) will be compatible: i.e. code for 0.2.0 should run with Lexon 0.2.1 but NOT vice versa. This is only a rule of thumb though and the cost of staying always true to this convention might sometimes be forbidding. After Lexon 1.0, no compatibility breaking changes are expected until Lexon 2.0.
The version number can currently be completely free form, it is not automatically processed yet but only an indication to users.

GRAMMAR

This is a word that is part of the meta structure of Lexon code. It groups and gives meaning to the terms following it.

EXAMPLE

```
LEXON: 0.2.9
```

LEXON 0.2

MAKE A PAYMENT

send tokens

<person> makes a payment [to escrow] <comparison>

CRYPTO

This term articulates the sending and receipt of crypto currency and any type of token.

VARIANTS

MAKES A PAYMENT

LEXON 0.6

EXAMPLE

If the Owner makes a Payment to Escrow equal to the Current Tax, then the Next Tax Due Date is redefined to be the time of Tax Frequency later than it was before.

MAY

permission to specific acting person

<person> may <statement>

Only the person named before 'may' can initiate the performance of what is described in a clause. No one can initiate the performance of a clause that is not lead in with a 'may', except by reference from another clause. This is the main organizational element to assign rights in a contract.

PERMISSION

This keyword allows to articulate who should be allowed to initiate the performance of a given clause.

LEXON 0.1

EXAMPLE

The Executor may pay the Inheritance to the Heir.

MINUS

subtract

<value> minus <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.2

EXAMPLE

The "Last Voting Phase Period Number" be defined as the sum of the Starting Period Number and the Voting Phase Duration In Periods minus 1.

MUST

verification condition

<name> must [be] <value>

VERIFICATION

This term allows for formal verification, enforcing that certain names may have only certain values under certain conditions. Performance of code stops when a verification fails.

LEXON 0.3

EXAMPLE

```
The Vote must be "yes"
or "no".
```

NEVER

disallow a value

never <value>

never <type>

VERIFICATION

This term allows for formal verification, enforcing that certain names may have only certain values under certain conditions. Performance of code stops when a verification fails.

LEXON 0.6

EXAMPLE

"Owner" is a person, and never no-one.

NO

binary

no

VALUE

This is a value that a name can be defined to have or that the meaning of a name can be compared against. It can also be used to articulate verifications.

LEXON 0.3

EXAMPLE

```
The Vote must be "yes"
or "no".
```

NO-ONE

enforce a person to be appointed

no-one

VALUE

This is a value that a name can be defined to have or that the meaning of a name can be compared against. It can also be used to articulate verifications.

LEXON 0.6

EXAMPLE

"Owner" is a person, and never no-one.

NOT

inverse binary

not <binary>

COMPARISON

This is an comparison operator to compare two values. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

LEXON 0.2

EXAMPLE

```
Also provided that the
Owner is not This
Contract.
```

NOT THE CASE

comparison of binary to true

not the case [that] <binary>

TEST

This is an comparison operator that operates on a binary value. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

LEXON 0.3

EXAMPLE

CHAPTER: Pay Back.
The Client may if Due Date is past and it is not the case that Provision of Services Have Met the Defined Service Criteria pay the Service Fee from escrow to themselves, and also pay the Fee from escrow to the Assessor.

NOTIFY

record a result

notify <person>
<person> be notified.

OUTPUT

This is a keyword that signals a result that the performance of the code should share with the world: e.g. print to the screen or write to the blockchain receipt log. Results can often be encrypted to facilitate controlled sharing of private information.

EXAMPLE

```
The Exception Handler
be notified.
```

VARIANTS

BE … NOTIFIED

SYNONYMS

SEND A NOTIFICATION

LEXON 0.3

NOW

point in time

now

VALUE

This is a value that a name can be defined to have or that the meaning of a name can be compared against. It can also be used to articulate verifications.

EXAMPLE

Next Tax Due Date be redefined as the time given in Tax Frequency after now.

LEXON 0.2

NUMBER OF

optional indicator that the value and not the name is used for a calculation

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.3

EXAMPLE

```
increase the count of No
Votes by the number of
the Shares of the
Member.
```

NUMBER OF DAYS

optional indicator that the value and not the name is used for a calculation

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.6

EXAMPLE

At all times provided that Next Tax Due Date is not further in the future than the number of days given as Tax Frequency.

OFFER

send tokens

<person> [may] offer[s] [a] <amount>

CRYPTO

This term articulates the sending and receipt of crypto currency and any type of token.

LEXON 0.3

EXAMPLE

An Applicant may offer a Token Tribute in Approved Tokens, set the Shares Requested, set the Details, and by this Create a Proposal with the Shares Requested.

ON

comparison of binary to true

<binary> on

TEST

This is an comparison operator that operates on a binary value. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

LEXON 0.3

EXAMPLE

```
Provided there is no
Exceptional Status on.
```

OR

options

<binary> or <binary>

<type> or <type>

COMPARISON

This is an comparison operator to compare two values. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

LEXON 0.1

EXAMPLE

"Possessor" is a person, or no-one.

PAID

optional reaffirmation that an amount has been paid in prior

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.6

EXAMPLE

```
Refund the paid Amount
from Escrow to the
Purchaser.
```

PASSED

earlier in time

<point in time> [has] passed

COMPARISON

This is an comparison operator to compare two values. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

EXAMPLE

```
Anyone may: if Next Tax
Due Date has passed,
then:
Trigger The Default.
```

VARIANTS

HAS PASSED

LEXON 0.2

PAST

earlier in time

<point in time> [is] past

COMPARISON

This is an comparison operator to compare two values. It operates on names and literal values. Comparisons are used to assigned the binary result to names, to inform the performance of the code and to evaluate verification claims.

LEXON 0.2

EXAMPLE

```
If Time for Next Payment
is now or past then
pay an Installment to
themselves,
and afterwards if the
escrow is 0 then
terminate this contract
else increase the Time
for Next Payment by the
Time between Payments.
```

PAY .. TO

send tokens

<person> pays <amount> to <person>.

Send or receive crypto currency, depending on the receiver. 'Escrow' is the described smart contract system itself.

CRYPTO

This term articulates the sending and receipt of crypto currency and any type of token.

EXAMPLE

The Grantor pays an Inheritance into Escrow.

VARIANTS

PAYS .. TO / PAYS IN / PAYS .. INTO ESCROW / IS PAID INTO ESCROW

SYNONYMS

SEND, RETURN

LEXON 0.2

PERSON

a blockchain account or address

"<name>" is a person.

TYPE

This is a type name, which is used to clarify what kind of data which name definition might stand for. A type name can be used like a definition, i.e. a name could be defined that has the same name like its type. But no definition may have the name of a type other than its own.

LEXON 0.1

EXAMPLE

"Grantor" is a person.

PLUS

add

<value> plus <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.2

EXAMPLE

t0 plus t2 to the power of 2 divided by t2.

POWER OF

mathemical exponent

<value> to [the] power of <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.2

EXAMPLE

t0 plus t2 to the power of 2 divided by t2.

PREVIOUS

prior value of a name

previous <name>

REFLECTION

This keyword allows to reach back to previous definitions of a name.

LEXON 0.6

EXAMPLE

```
Increase the count of
Cases by the amount of
previous Attempts.
```

PRIOR

prior value of a name

prior <name>

REFLECTION

This keyword allows to reach back to previous definitions of a name.

LEXON 0.6

EXAMPLE

```
Decrease the number of
Tokens by the prior
balance.
```

PRODUCT OF ... AND

result of multiplication

product of <value> and <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.3

EXAMPLE

The "Cost" be redefined as the product of the former Cost and the EPS.

PROVIDED

start of a verification condition of a clause

provided <binary>

VERIFICATION

This term allows for formal verification, enforcing that certain names may have only certain values under certain conditions. Performance of code stops when a verification fails.

LEXON 0.3

EXAMPLE

```
Provided there is no
Exceptional Status on.
```

PUBLISH

record a result

publish <name>

OUTPUT

This is a keyword that signals a result that the performance of the code should share with the world: e.g. print to the screen or write to the blockchain receipt log. Results can often be encrypted to facilitate controlled sharing of private information.

LEXON 0.3

EXAMPLE

```
Set Next Tax Date,
afterwards Update
Records,and afterwards
publish Owner and Cost.
```

RECORD

optional reaffirmation that the invocation of a clause will change the state of a contract

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.3

EXAMPLE

```
Increase the count of Yes
Votes by the number of
the Shares of the
Member,
and also record the Last
Vote of the Member,
and also Track Maximum
of Total Yes Votes.
```

RECORDED VALUE

optional indicator that the current value of a definition is used for a calculation

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.3

EXAMPLE

```
If the count of Total
Shares is greater than the
recorded value of
Maximum Total Shares
At Yes Vote then
the recorded value of
Maximum Total Shares
At Yes Vote be changed
at that point to the count
of Total Shares.
```

REDEFINE

change the meaning of a name

"<name>" is redefined to be <value>.

Used to change the meaning of a name.

Define works only for names that did not have a meaning assigned before. To change a name's meaning, use 'redefine' or 'change'.

Good style avoids redefinitions as they can degrade readability of code.

ASSIGNMENT

This term is used to define the value of a name.

VARIANTS

REDEFINES / REDEFINED / REDEFINED TO BE

SYNONYMS

CHANGE

LEXON 0.2

EXAMPLE

"Due Date" is redefined to be Tomorrow.

REMAINDER

remainder of division or escrow

remainder of the escrow
remainder of <division>

1. Remainder of a division of whole numbers.
2. Remainder of the amount in escrow.

MATH / CRYPTO

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims. / This term articulates the sending and receipt of crypto currency and any type of token.

LEXON 0.2

EXAMPLE

```
Return the remainder of
the escrow to the Payer.
```

RESULTING

optional filler indicating that a name is re-defined by a calculation

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.3

EXAMPLE

The "Voting Phase Periods" is defined as the whole number resulting from dividing the Voting Phase Duration by the Period Duration.

RETURN

send tokens back

return <amount> to <person>

Send received crypto currency back to the sender.

CRYPTO

This term articulates the sending and receipt of crypto currency and any type of token.

SYNONYMS

PAY

LEXON 0.2

EXAMPLE

Return the Payment to the Customer.

REVEAL

record a result

reveal <value> to <person>

OUTPUT

This is a keyword that signals a result that the performance of the code should share with the world: e.g. print to the screen or write to the blockchain receipt log. Results can often be encrypted to facilitate controlled sharing of private information.

LEXON 0.2

EXAMPLE

```
The Owner may reveal
Data and Time to a
Viewer.
```

REVOKE

set binary false

revoke <binary>

ASSIGNMENT

This term is used to define the value of a name.

LEXON 0.2

EXAMPLE

```
CLAUSE: End
Exceptional Status.
If in Exceptional Status,
the Exception Handler
may:
revoke the Exceptional
Status.
```

SECONDS

time duration

seconds

UNIT

This is a unit for values, it can also be used to mean 1 unit of this.

LEXON 0.3

EXAMPLE

The "Period Duration" is set to a duration of time in seconds, greater than 0.

SECTION

non-processing headline

SECTION: <headline text>

A headline comment that has no meaning for the program execution and is ignored.
It is used to signify a logical part in the code without having any binding meaning for the performance. It can be used entirely freely to improve readability for humans.
Anything after the SECTION keyword is ignored for the contract performance until the next line break. Dots are assumed to be part of a comment. The section headline text can be multiple lines on a printout or on-screen if the program editor or the word editor chose to display text wrapped across multiple lines. The section headline text still ends only after the place where the enter key had been hit to start a new line.

GRAMMAR

This is a reserved keyword that has significant meaning to the Lexon transpiler and virtual machine. It cannot be used as a variable or contract name. It does not matter whether any of its letters are written in lower or upper case. But other words can include this keyword as part.

EXAMPLE

```
SECTION: TAX.
```

LEXON 0.2

SEND A NOTIFICATION

record a result

send a notification to <person> to <message>

OUTPUT

This is a keyword that signals a result that the performance of the code should share with the world: e.g. print to the screen or write to the blockchain receipt log. Results can often be encrypted to facilitate controlled sharing of private information.

EXAMPLE

Afterwards, send a Notification that a Default occurred, and send a Notification to the Possessor to send the Notebook to the Owner.

SYNONYMS

NOTIFY

LEXON 0.6

SUBTRACT FROM

subtraction

subtract <value> from <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.2

EXAMPLE

```
subtract test var1 from
test var2.
```

SUM OF

result of addition

sum of [all] <name> <contractor>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.2

EXAMPLE

```
CLAUSE: Payout.
Pay the Inheritance times
the Share
divided by the sum of all
Shares to the Heir,
and thereby terminate
this contract.
```

TENTH

result of division by ten

tenth [of] [a] <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.3

EXAMPLE

"Wait" is defined as a duration of a tenth of a day.

TERMINATE

end the performance

<person> may terminate this contract.

No contract clause can be performed after this statement has been met. The original creator can access any remaining balance.

PERFORMANCE

This term controls the performance of the code, e.g. if following statements apply, or not; or if the contract should terminate.

LEXON 0.2

EXAMPLE

```
The Executor may pay
the Escrow to the Heir
and thereby Terminate
this contract.
```

TERMINATE ALL CONTRACTS.

partly end performance

PERFORMANCE

This term controls the performance of the code, e.g. if following statements apply, or not; or if the contract should terminate.

LEXON 0.2

EXAMPLE

```
The Executor may
for all Heirs Payout,
and thereby terminate all
contracts.
```

TERMS

start of the optional code section that contains statements that are relevant across multiple contracts.

TERMS:

Code after TERMS and until the CONTRACTS keyword appears, is understood to be relevant across many different contracts. Those contracts are then defined under the CONTRACTS section.

The code after TERMS will usually start with definitions, followed by preparatory instructions and then clauses. All of this is optional and the TERMS keyword can be left out when there is no CONTACTS keyword below.

GRAMMAR

This is a word that is part of the meta structure of Lexon code. It groups and gives meaning to the terms following it.

LEXON 0.2

EXAMPLE

```
TERMS:
```

TEXT

a value that containts a word, a text, letters or numbers

"<name>" is [a] text.

This type is used to state that a variable will contain a blockchain address, or a blockchain account, which is the way that ids exist on a blockchain.

TYPE

This is a type name, which is used to clarify what kind of data which name definition might stand for. A type name can be used like a definition, i.e. a name could be defined that has the same name like its type. But no definition may have the name of a type other than its own.

LEXON 0.2

EXAMPLE

```
The "Text" is a text.
```

THE

optional article to increase readability

(none)

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

SYNONYMS

A, AN

LEXON 0.1

EXAMPLE

```
The Executor may pay
the Escrow to the Heir
```

THEMSELVES

pronoun

<person> pay <amount> to themselves.

A pronoun that means the subject of the statement, a person.

PRONOUN

This is a reserved keyword that has significant meaning to the Lexon transpiler and virtual machine. It cannot be used as a variable or contract name. It does not matter whether any of its letters are written in lower or upper case. But other words can include this keyword as part.

LEXON 0.2

EXAMPLE

The Arbiter may pay from escrow the Fee to themselves.

THEN

conditional and chronological order

<statement>, [and] then <statement>

SEQUENCE

This word puts statements before and after it in chronological order. This is a reserved keyword that has significant meaning to the Lexon transpiler and virtual machine. It cannot be used as a variable or contract name. It does not matter whether any of its letters are written in lower or upper case. Other words can include this keyword as part.

LEXON 0.2

EXAMPLE

```
The Holder may pay the
Amount into escrow,
and then the Bet is
deemed Closed.
```

THERE IS

optional wording to increase readability of conditions

there is <condition>

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.6

EXAMPLE

```
Provided there is no
Exceptional Status on.
```

THEREBY

conditional and chronological order

<statement> thereby <statement>

indicates that the following statement should come into effect if the preceding was performed. If the preceding statement could not be performed, the following statement should not even be looked at.

SEQUENCE

This word puts statements before and after it in chronological order. This is a reserved keyword that has significant meaning to the Lexon transpiler and virtual machine. It cannot be used as a variable or contract name. It does not matter whether any of its letters are written in lower or upper case. Other words can include this keyword as part.

EXAMPLE

```
The Executor may pay
the Escrow to the Heir
and thereby terminate
this contract.
```

VARIANTS

AND THEREBY

SYNONYMS

AFTERWARDS, AND ALSO

LEXON 0.2

THIRD

result of division by three

third [of] [a] <value>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.3

EXAMPLE

"Cool Down" is defined as a duration of a third of a day.

THIS

optional demonstrative determiner to increase readability

(none)

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.3

EXAMPLE

```
Terminate this Will.
```

THIS CONTRACT

the address of this code when instantiated on the blockchain

this contract

This definition is always available and stands in for the blockchain identity of the code itself that it is part of. It can be used to define a name that when used makes the code more readable.

VALUE

This is a reserved term that has significant meaning to the Lexon transpiler and virtual machine. It cannot be used as a definition or contract name. It does not matter whether any of its letters are written in lower or upper case. Other words can include this term as part.

LEXON 0.2

EXAMPLE

```
"Will" is defined as this
contract.
```

TIME

a date and time to the second

"<name>" is [a] time.

This keyword is used to state that a name means a timestamp, consisting of a date and a time with precision of one second.

TYPE

This is a type name, which is used to clarify what kind of data which name definition might stand for. A type name can be used like a definition, i.e. a name could be defined that has the same name like its type. But no definition may have the name of a type other than its own.

LEXON 0.1

EXAMPLE

```
"Deadline" is a time.
```

TIME PASSED SINCE

the time span between now and a given time

time passed since <point in time>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.3

EXAMPLE

CLAUSE: Current Period Number.
The "Current Period Number" is defined as the whole number resulting from calculating the time passed since Summoning Time, divided by the Period Duration.

TIMES

multiplication

<name> times <number>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.3

EXAMPLE

```
CLAUSE: Compensation.
Rage Compensation is
defined as
the amount in escrow
times the Burned Shares
divided by the sum of
the Total Shares and the
Burned Shares.
```

TO

optional preposition to increase readability

(none)

FILLER

This is a filler that is ignored by the Lexon transpiler and virtusal machine. It can be ommitted with no change to script functionality. The legal prose output, however, could be affected.

LEXON 0.1

EXAMPLE

```
The Grantor appoints a
Person to be Executor.
```

TOKEN TYPE

the name of the currency or token to be used

TYPE

This is a type name, which is used to clarify what kind of data which name definition might stand for. A type name can be used like a definition, i.e. a name could be defined that has the same name like its type. But no definition may have the name of a type other than its own.

LEXON 0.3

EXAMPLE

The Summoner sets the "Approved Token" to any token type.

TRANSFER

send tokens

<transfer> <amount> [from escrow] to <person>

CRYPTO

This term articulates the sending and receipt of crypto currency and any type of token.

LEXON 0.3

EXAMPLE

Transfer the Token Tribute in Approved Tokens from escrow to Guild Bank.

UNDEFINED

value for 'no value'

undefined

VALUE

This is a value that a name can be defined to have or that the meaning of a name can be compared against. It can also be used to articulate verifications.

LEXON 0.6

EXAMPLE

"Last Status Claimed" is a text, or undefined.

WHOLE NUMBER

discarding decimal fractions of a number

whole number [resulting from] <formula>

MATH

This is an algorithmic operator to calculate results from given values. Its operators can be names and literal values. Such operations are used to calculate values to be assigned to names, to make comparisons that will inform the performance and to evaluate verification claims.

LEXON 0.3

EXAMPLE

The "Voting Phase Periods" is defined as the whole number resulting from dividing the Voting Phase Duration by the Period Duration.

WITH

list values for a clause

<clause> with <name>

ASSIGNMENT

This term is used to define the value of a name.

LEXON 0.2

EXAMPLE

The Summoner enters into a Member Contract with Summoner's Initial Number Of Shares.

YES

binary true

yes

VALUE

This is a value that a name can be defined to have or that the meaning of a name can be compared against. It can also be used to articulate verifications.

LEXON 0.3

EXAMPLE

```
The Vote must be "yes"
or "no".
```