Lexon

plain-text programming

color off

Vocabulary

You don't need this to read and understand a Lexon text. This page is for getting an idea how to write one.

This is the word list of Lexon 0.3. The page is a learning tool to help grasp the bigger picture, not neccessarily to memorize individual words. To this end, one thousand links are at your disposal below for fast navigation between words, examples and references. You will find that clicking around reveals the deeper structure of Lexon, beyond words.

The point of Lexon is that one does not have to learn it to read it. Learning to *write* Lexon is best done by looking at examples rather than memorizing words. This is no different than how a human language, or a programming language, or even legalese is picked up. The allowed use of a word, its context, are what matters. As the Lexon grammar is a so-called *controlled grammar*, only some ways of using a word are permitted, fewer than in normal English. Studying this is better learned by doing than done by learning. In fact, it is particularly hard to 'unlearn' what you know is correct in normal language. This is why the best way to use this word list is to compare and explore, and focus on the examples, skipping from one to the next by clicking on any of its words. You will – and that is a unique quality of Lexon – get a feel for what works.

- ⟶ Words
- ⟶ Examples

Breadth and Capacity

Lexon's vocabulary is unlimited because a) any noun or compound term can be defined and used in a Lexon text, b) any phrase can be used as clause name and then used as part of a sentence elsewhere in the text, and c) the Lexon compiler is extensible and keywords can and are added while the grammar grows more powerful.

a) *Definitions*:
see Payer, Payee, etc. in the ⟶ escrow example.

b) *Clauses*:
see Noticed, Factually Breached, etc. in the ⟶ evaluation license example.

c) *Keywords*:
Cf. vocabulary 0.3 below vs. ⟶ 0.2 of the 2020 lexon bible.

The latter is a slow and incremental process. A special, faster process has been prepared for verbs

of foundational importance – like *move* for robotics – that can sustain an entire domain. The two former points are instant and happen when a user authors a Lexon text. Each Lexon digital contract therefore has its own vocabulary, extending the dictionary on the fly while drafting; as integral and organic part of the writing process. This is in keeping with the usual way that paper contracts are written: terms are being defined for clarity, laying the ground for an agreement's text. That's exactly how Lexon's vocabulary grows while penning a digital contract.

However, as it comes, before any definitions are added, Lexon 0.3 understands 91 keywords plus variations, of which roughly half are processed in an interesting way. Many have only one specific function as marker, like ⟶ **clause**, or make the list solely by virtue of being part of a fix multi-word term, with no independent function, like ⟶ **off** in **sign off**.

To put the word count into perspective, a modern 3rd generation programming language like Rust has about 50 reserved keywords, which are mostly used in a rigid, less interesting way. Beginner's English is said to consist of about 300 words, the Basic English world language project [1] has about 850. These latter counts include many nouns; Lexon's vocabulary contains almost no nouns because these are as a rule defined by the writer of a Lexon text and the Lexon compiler understands them from their function as implied by the rest of the text. This is a fundamental aspect of Lexon's approach and the reason why the Lexon compiler needs to 'understand' relatively few words out of the box. The nouns and phrases that a user adds are inevitably what gives a text depth. The way that definitions work, the compiler learns the role of the new words on the go and recognizes them in the text from that point on. Clause names, however, can be the most interesting because they are the way to insert any complexity of grammar, which can make Lexon texts look rich and elegant. This is a powerful design choice that serves to include language constructs outside of the limit up to which the controlled grammar has to be observed – offering instead of a hard stop, freedom to be fully creative within the safety of a well defined reference frame. Lexon 'understands' such phrases (the clause names) en block, obviously, because the clause itself defines their meaning. The (multi-word) name of a clause's internal grammatical composition is not analysed. The way that the processing of clause names is isolated from the rest of the text is making it possible to mix these monolithic elements with the bulk of the text that the Lexon compiler processes word-for-word, i.e., 'more truely understands'. This combination is a pragmatic way to empower the users to add complexity without having to think about any controlled grammar rules, but importantly also: to freely add vocabulary. The new words are baked into the specific grammatical way that they are presented (in the clause name). They need not be explained further. This mode of extension makes perfect sense for Lexon. 3rd generation programming languages generally allow to add variable names and add types. But they can never reach beyond their fix grammar. Lexon, in contrast, accepts any grammatical extension through the freedom it offers in naming clauses. In this, there is not even a requirement to avoid the initial 91 words that Lexon knows when creating compound new compositions.

But the list of initial words cannot be redefined. It is therefore on the one hand desirable that it includes few nouns. The reason that Lexon, on the other hand, has a larger basic vocabulary than, e.g., Rust is that Lexon's grammar is designed to enable multiple ways of expressing the same meanning, to make writing more intuitive, and to allow for more fluid, natural-appearing texts. This does not mean that Lexon's grammar is ambigious, i.e., that the same sentence could have multiple meanings. It only means that the same meaning can be expressed by differently worded sentences. This is normal for any programming language, and only some developer communities have the ambition to adhere to a style at all times that defines the one, 'right' way for anything that could be expressed. This is always only a convention though.

[1] Basic English ⟶ *https://simple.wikipedia.org/wiki/basic_english*

Words

This is the list of the *known words* in Lexon 0.3 that between themselves stand up the Lexon grammar.

a after afterwards all also amount* an and announced appoint appoints as at author authors be been being binary* certified certifies certify clause comment comments contract contracts current data* day days declare declared declares defined equal equaling escrow filed file files fix fixed fixes for from general given grant grants has herself himself hour hours if in into is itself least lex lexon lies may millisecond milliseconds minute minutes month months myself no not now of off on oneself or ourselves passed past pay pays per person* preamble provided register registers remainder respective return returns second seconds send sends signed so terminate terminates terms text that the themself themselves then there therefor therefore these this time* to true was week weeks with year years yes yourself yourselves

In case you have to, these words can mostly be redefined as *names* but in that case can then not serve in their original function as described below anywhere in the contract. The exception are *category* names (types, marked with an asterisk (*) above). They cannot be redefined but can be used as generic names (see, e.g., ⟶ **amount**). All words can be used as *part* of names without losing their original, stand-alone function.

In the individual, per-word entries below, the first information about each word is, in what linguistic capacity it is used in Lexon. This angle can help because English words frequently cover two ore more different grammatical roles. The second line is technical, based on computer sciences designations for the function that a word is used for in Lexon. It will more often than not be a helpful pointer for non-programmers, too. In many cases, a description follows that describes the use of the word and occasionally presents additional context. Note that the description often seems to explain the obvious, because you know how English works. However, it can help to spell it out to learn to write Lexon. After that, one or more examples show the word in the context of an examples sentence. This sentence can be inspected in the context of a complete digital contract by clicking the link immediately below the example sentence.

---

word

linguistic function
`programming function`

Description.

Example sentence.

---

see ⟶ complete example contract

The html version of this page can be switched to colored code. The style for different words in different roles is:

Lexon keywords

entry's keyword

user-defined terms

literal numbers

meta information

Alphabetical Entries

The entries are based on the grammar version *0.2.20 / subset 0.3.8 alpha 79 - English / Reyes*.

a, an

indefinite article
`no op`

Articles can be left out with no change in meaning. They are optional to increase readability.

They can be omitted, because the name they precede must always be unambigous on its own. This is familiar practice with paper contracts.

Same goes for ⟶ **the**, ⟶ **this**, ⟶ **these**.

> *The Payer pays **an** Amount into escrow, appoints the Payee, appoints the Arbiter, and fixes the Fee.*

see ⟶ escrow.lex

after

timewise prepostion
`time operator`

After is used to calculate a point in time, relative to a given one.

> *"Termination Period" is defined as 365 days **after** the Termination Statement Time.*

see ⟶ statement.lex

For more on how to use time, see ⟶ **hours** and ⟶ **days**.

afterwards

adverb
`causal concatenation`

Keyword that introduces temporal order, which is not a default in Lexon.

Separate sentences are performed independently of each other, *declaratively*, rather than one after the other. **Afterwards** serves to bind statements into one sentence and to establish that the phrase following it is performed only after all side effects of the phrase before it have been established.

To illustrate by example, in the Lexon sentence given below, the ⟶ **remainder** is what remains after the Fee mentioned before **afterwards** has been deducted.

Cf. ⟶ **therefore**.

> *The Arbiter may pay from escrow the Fee to themselves, and **afterwards** return the remainder of the escrow to the Payer.*

see ⟶ escrow.lex

all

adjective
`quantifier`

Only with ⟶ **contracts**.

All contracts means all digital contracts in a contract system. This includes the main contract as well as the covenants, or subcontracts.

also

adverb
`no op`

Only appears with ⟶ **and**.

And also is synonymous to ⟶ **and**.

amount

noun
`type`

Defines that a name stands for an amount. In the example, Digital Asset Collateral is marked as being used as the handle for a specific number in the document.

> *"Digital Asset Collateral" is an **amount**.*

see ⟶ statement.lex

Amount can also be used as a name itself, without being first defined. It can only stand for an amount – i.e., for a number and not a text or a time – and **Amount** must be spelled with a capital 'A' in this case.

> *The Payer pays an **Amount** into escrow, appoints the Payee, appoints the Arbiter, and fixes the Fee.*

see ⟶ escrow.lex

and

conjunction
`logical and procedural operator`

Concatenates actions ...

> *The Payer pays an Amount into escrow, appoints the Payee, appoints the Arbiter, **and** fixes the Fee.*

see ⟶ escrow.lex

... as well as logical expressions.

A phrase that contains **and** is ⟶ **true** if the part left and the part right of the **and** are true. There can also be multiple parts, each separated by **and**. All of them need to be true for the entire expression to be true.

> *"Factually Breached" is defined as: this License is Commissioned **and** the Comment Text is not fixed, or this License is Published **and** there is no Permission to Comment **and** the Notice Time lies at least 24 hours in the past.*

see ⟶ evaluation.lex

For precedence and the interplay between **and** and **or**, see ⟶ **or**.

announced

adjective
`truth value`

Functions like ⟶ **fixed**.

appoint , appoints

verb
`parameter assignment operator`

Expresses that the subject of the sentence will determine what the specified object's names will mean concretely. In the example, who the Payee and the Arbiter are.

Functions like ⟶ **fix**, see additional notes regarding the *subject* there.

> *The Payer pays an Amount into escrow, **appoints** the Payee, **appoints** the Arbiter, and fixes the Fee.*

see ⟶ escrow.lex

Functional synonym to ⟶ **certify**, ⟶ **declare**, ⟶ **file**, ⟶ **fix**, ⟶ **grant**, and ⟶ **register**.

as

conjunction
`value assignment operator part`

Assigns the value of the expression to its right to the name on its left.

> *The Secured Party may file a Termination Statement, and certify the Termination Statement Time **as** the then current time.*

see ⟶ statement.lex

As can also make a name ⟶ **true** that was defined as a ⟶ **binary**. In this example, License serves as an object that means the entire contract system, which ultimately is a redundant scope. The relevant mutation is that Commissioned becomes a fact, i.e., ⟶ **true**.

> *The Licensor may certify this License **as** Commissioned.*

see ⟶ evaluation.lex

at

preposition
`quantifier`

Only in conjunction with ⟶ **least**.

> *"Factually Breached" is defined as: this License is Commissioned and the Comment Text is not fixed, or this License is Published and there is no Permission to Comment and the Notice Time lies **at** least 24 hours in the past.*

see ⟶ evaluation.lex

author, authors

noun
`keyword`

The information after the **AUTHOR(S)** keyword is expected to be the name(s) of the creator(s) of the Lexon text. They are meta data, not parsed, and not used in the document itself.

As a convention, **AUTHOR** and **AUTHORS** are usually spelled in uppercase.

> ***AUTHORS**: FLORIAN IDELBERGER, HENNING DIEDRICH*

see ⟶ evaluation.lex

be

verb
`assignment`

Used with a meaning like *shall*.

Functions like ⟶ **is** and can be used synonymously. The linguistic difference is irrelevant for the machine.

> *"Noticed" **be** defined as a Notice Time being fixed.*

been

verb
`comparison operator`

Appears only in conjunction with ⟶ **has**.

Has been functions like ⟶ **being**.

> *The Arbiter may, if the Notice Time has **been** fixed, return the Fee to the Seller.*

being

present participle
`comparison operator`

Compares the expression to its left with the expression on its right and results in everything together being ⟶ **true** or ⟶ **false**.

In the example, **being** tests Notice Time for whether it had been ⟶ **fixed** before. Noticed will be true exactly when Notice Time is known, and false if, no value has been given for Notice Time before at any point during the lifetime of the contract.

> *"Noticed" is defined as a Notice Time **being** fixed.*

see ⟶ evaluation.lex

binary

adjective
`type`

Defines a name as standing for a binary value, e.g., ⟶ **yes** or ⟶ **no**, or ⟶ **true** or ⟶ **false**.

Note that an undefined binary name is considered to have the value ⟶ **false**. ⟶ **declaring** a name sets it to ⟶ **true**. Likewise, testing whether a binary name is ⟶ **declared**, checks whether it is ⟶ **true**.

> *"Default" is a **binary**.*

see ⟶ statement.lex

certified

adjective
`«defined»`

Expresses that a name has a value assigned, i.e., is not *unbound* or *undefined*.

In the example, ⟶ **being** tests Notice Time for whether it had been **certified** before. Noticed will be ⟶ **true** exactly when a Notice Time is known, and ⟶ **false** if no value has been given for Notice Time before, at any point during the lifetime of the contract.

Functions like ⟶ **fixed**.

> *"Noticed" is defined as a Notice Time being **certified**.*

certifies, certify

verb
`assignment operator`

Expresses that the subject of the sentence will determine what the specified object's names will mean concretely. In the example, who the Payee and the Arbiter are.

Functions like ⟶ **fix**, see additional notes regarding the *subject* and invocation there.

> *The Filing Office may **certify** the File Number.*

see ⟶ statement.lex

Functional synonym to ⟶ **appoint**, ⟶ **declare**, ⟶ **file**, ⟶ **fix**, ⟶ **grant**, and ⟶ **register**.

clause

noun
`function keyword`

Signals the start of a clause. A colon must follow, and the name of the clause. Then, the statements that constitute the clause.

Almost every digital contract has one or more clauses. Only in rare, simplistic cases does a contract have only a *recital*.

Either a clause's name is used to instigate actions that change the state of the contract:

> ***CLAUSE**: Pay Back.*
>
> *The Arbiter may pay from escrow the Fee to themselves, and afterwards return the remainder of the escrow to the Payer.*

see ⟶ escrow.lex

Or, a clause name can itself be a value, if the clause uses ⟶ **defined**.

The concrete meaning of the name of such clause is dynamic. That is, the concrete meaning of the clause name is not assigned once and for all at any point in time. Instead, whenever the clause name is used elsewhere in any context, the expression right-hand of ⟶ **defined** is re-evaluated for its now current result, which is then the meaning, or value, of that clause name.

> ***CLAUSE**: Termination Period.*
>
> *"Termination Period" is **defined** as 365 days after the Termination Statement Time.*

see ⟶ statement.lex

The clause name can be used as an expression in the context of other clauses, i.e the name can be used like a value.

The example below uses the name Termination Period that is defined in the example above.

> *The Filing Office may, if the **Termination Period** has passed, terminate this contract.*

see ⟶ statement.lex

comment, comments

noun
`comment keyword`

Start of comments that are not translated by the compiler.

Functions like ⟶ **preamble** but can be used multiple times in different places.

Lexon is self-documenting, which greatly diminishes the role of comments. They should be used sparingly or not at all. They can help to explain a more convoluted set of conditions, as can be found in contracts that need to spell out things in detail, including all relevant fringe cases.

Care should be taken to clarify that a comment is not part of the legally binding text; but is written to provide motivation or explain complex aspects with a broad brush, to make the contract easier to understand for a human reader. Such clarification may be added as part of the comment itself.

As a convention, **COMMENT** is usually spelled in uppercase.

Cf. ⟶ **preamble**.

> ***COMMENT**: A license can be for any tangible or intangible good.*

contract, contracts

noun
`self reference`

Contract as well as ⟶ **all contracts** stand for the contract (system) itself, including all covenants (subcontracts)

Contract can either be used to define a proper name for the digital contract:

> *"Financing Statement" is this **contract**.*

see ⟶ statement.lex

Or, **contract** as well as ⟶ **all contracts** can be used as object to ⟶ **terminate**.

> *The Filing Office may, if the Termination Period has passed, terminate this **contract**.*

see ⟶ statement.lex

current

adjective
`time value`

Only appears with ⟶ **time**.

> *The Filing Office may fix the Initial Statement Date as the **current** time.*

see ⟶ statement.lex

data

noun
`type`

Defines a name as standing for a piece of data.

Data can be a text, a number, a hash, a blockchain address, or an id of any type.

> *"File Number" is data.*

see ⟶ statement.lex

day, days

noun
`time unit`

Used to describe a duration.

A duration can be used to calculate a point in time relative to another point in time. For example, relative to ⟶ **now** or ⟶ **in the past**, or – as in the example below - relative to a name that means a specific time.

> *"Termination Period" is defined as 365 **days** after the Termination Statement Time.*

Cf. ⟶ **hours**.

declare, declares

verb
`truth assignment operator`

Used to state that something has happened, or is ⟶ **true**.

Technically, **declare** assigns the truth value, ⟶ **true**, to a name. That name must have been defined (see ⟶ **is**) as a ⟶ **binary**.

In the example this means that Default is now true. Note that before that, it was ⟶ **false**.

Cf. ⟶ **binary**.

> *The Secured Party may **declare** Default.*

see ⟶ statement.lex

declared

adjective
`«true»`

Synonym to ⟶ **true**.

In the example, the fact that Default has been **declared** is the same as saying that it is ⟶ **true** that Default happened.

> *The Filing Office may, if Default is **declared**, pay the Digital Asset Collateral to the Secured Party.*

see ⟶ statement.lex

defined

adjective
`assignment operator`

Always used with ⟶ **is**, or ⟶ **be** and ⟶ **as**, to describe the meaning of the name to its left by means of the expression on its right.

The meaning is dynamic. That is, the concrete meaning is not assigned once and for all at any point in time. But instead, whenever the name that is being defined is used elsewhere in any context, the expression right-hand of **defined** is re-evaluated for its now current result, which is in that moment the meaning of that name.

> *CLAUSE: Termination Period*
>
> *"Termination Period" is **defined** as 365 days after the Termination Statement Time.*

see ⟶ statement.lex

This type of sentence is the essence of a particular type of ⟶ **CLAUSE** whose name can be used like an expression, i.e., the name of such clause can be used like a value in the text of another clause.

The example below uses the name Termination Period that is defined in the example above.

> *The Filing Office may, if the **Termination Period** has passed, terminate this contract.*

see ⟶ statement.lex

equal

comparison operator
`equivalence of values`

Forms an expression that is ⟶ **true** if the values left and right of **equal** are the same.

Near synonym of ⟶ **equaling**.

> *"Parity" is defined as the Count of X being **equal** to the Count of Y.*

equaling

comparison operator
`equivalence of values`

Forms an expression that is ⟶ **true** if the values left and right of **equaling** are the same.

Near synonym of ⟶ **equal**.

> *"Parity" is defined as the Left Side **equaling** the Right Side.*

escrow

noun
`system variable`

The internal token escrow of a digital contract.

It is mostly used as object to the predicate ⟶ **pay**.

Using it with ⟶ **remainder** results into a number: the amount of tokens left in the escrow at that point in time.

> *The Arbiter may pay from **escrow** the Fee to themselves, and afterwards pay the remainder of the **escrow** to the Payee.*

see ⟶ escrow.lex

filed

adjective
`«defined»`

Asking whether a name **is filed** constitutes an expression that is ⟶ **true** in case the name had a value assigned to it previously. The expression is ⟶ **false** if the name had not been defined before during the lifetime of the contract.

> *... the Continuation Statement is **filed** ...*

see ⟶ statement.lex

To clarify, it does not matter if there is text somewhere in the contract that gives a name a concrete meaning. What matters is whether for a specific, live contract between concrete parties and with a concrete state, it so happened that it is clear what a specific name stands for, or, that what the name stands for exists.

If you take this example ...

> *The Filing Office may, if the Continuation Statement is **filed**, fix the Continuation Statement Date.*

see ⟶ statement.lex

... the phrase *the Continuation Statement is filed* is *true*, if what is described in the clause shown below ever happened. Concretely, if the Secured Party has filed the Continuation Statement.

> *CLAUSE: File Continuation.*
>
> *The Secured Party may file the Continuation Statement.*

see ⟶ statement.lex

Note that in this example contract, the Continuation Statement is defined as a binary. That means that it does not have any specific content beyond existing or not. The filing of it 'is' the statement that continuation is desired.

file, files

verb
`parameter assignment operator`

Synonym to ⟶ **fix**.

> *The Secured Party may **file** the Continuation Statement.*

see ⟶ statement.lex

fix, fixes

verb
`parameter assignment operator`

Indicates that the subject of the sentence will be who determines the meaning of the named objects.

Note that this cuts both ways. The subject might itself be determined by the act of fixing the objects: if it had not been settled yet who the name of the subject refers to, then whoever performs the fixing is from that point on named like the subject of this sentence. The name sticks for the remaining lifetime of the contract. Accordingly, in the example below, if the role of the Filer had not been determined, the person who is fixing the Filing Office etc. becomes the Filer. The way to prevent this automatism is to use ⟶ **may**.

The values that are assigned to the objects of the sentence are given by the subject when that person acts to invoke this clause.

> *The Filer **fixes** the Filing Office, **fixes** the Debtor, **fixes** the Secured Party, and **fixes** the Collateral.*

see ⟶ statement.lex

Functional synonym to ⟶ **appoint**, ⟶ **certify**, ⟶ **declare**, ⟶ **file**, ⟶ **grant**, and ⟶ **register**.

fixed

adjective
`«defined»`

Expresses that a name has a value assigned, i.e., is not *unbound* or *undefined*. In the example, ⟶ **being** tests Notice Time for whether it had been **fixed** before. Noticed will be true exactly when a Notice Time is known, and false if no value has been given for Notice Time before, at any point during the lifetime of the contract.

Functions like ⟶ **certified**.

> *"Noticed" is defined as a Notice Time being **fixed**.*

see ⟶ evaluation.lex

for

preposition
`no op`

Only in conjunction with ⟶ **file** or ⟶ **filed**.

The terms **file for** or **filed for** function like ⟶ **file** or ⟶ **filed** without **for**.

from

preposition
`transfer origin marker`

Only in conjunction with ⟶ **escrow**.

> *The Arbiter may pay **from** escrow the Fee to themselves, and afterwards return the remainder of the escrow to the Payer.*

see ⟶ escrow.lex

general

adjective
`no op`

Optional specification to ⟶ **TERMS**.

given

preposition
`conditional keyword`

Following statements are executed only if the immediately following condition is true.
Appears only together with ⟶ **that**.

given that is a synonym to ⟶ **if**.

> *The Filing Office may, **given** that the Continuation Window Start has passed, send the Notification Statement to the Secured Party.*

grant, grants

verb
`truth assignment operator`

Synonym to ⟶ **fix**.

> *The Licensee may **grant** the Permission to Comment.*

see ⟶ evaluation.lex

has

auxilliary verb
`part`

Only in conjunction with ⟶ **been** or ⟶ **passed**.

> *The Filing Office may, if the Continuation Window Start **has** passed, send the Notification Statement to the Secured Party.*

see ⟶ statement.lex

herself, himself

reflexive pronoun
`automatic reference`

Refers to the subject of the sentence.

Functions like ⟶ **themself**.

hour, hours

time unit
`time constant of 3600 seconds`

Used to describe a duration.

A duration can be used to calculate a point in time relative to another point in time. For example, relative to ⟶ **now** or ⟶ **in the past**.

> *"Factually Breached" is defined as: this License is Commissioned and the Comment Text is not fixed, or this License is Published and there is no Permission to Comment and the Notice Time lies at least 24 **hours** in the past.*

see ⟶ evaluation.lex

Cf. ⟶ **days**.

if

conjunction
`assertion`

Following statements are executed only of the immediately following condition is true.

The condition starts after **if** and ends with a comma, which can be followed by an optional ⟶ **then**.

The statements follow after that.

In this example ...

> *The Filing Office may, **if** the Continuation Window Start has passed, send the Notification Statement to the Secured Party.*

see ⟶ statement.lex

... the condition is:

> *the Continuation Window Start has passed*

... the statements are:

> *send the Notification Statement to the Secured Party*

in

preposition
`no op`

Only in conjunction with ⟶ **past**.

> *"Factually Breached" is defined as: this License is Commissioned and the Comment Text is not fixed, or this License is Published and there is no Permission to Comment and the Notice Time lies at least 24 hours **in** the past.*

see ⟶ evaluation.lex

into

prepositiion
`operator part`

Used with ⟶ **notify**, ⟶ **send** and ⟶ **pay**.

> *The Secured Party may pay a Reminder Fee **into** escrow.*

see ⟶ statement.lex

is

verb
`assignment and equality operator`

Can be used to define of what category a name is; to assign a value to a name; to compare a name to a value; or to check that something is the case.

> *"Payer" **is** a person.*

see ⟶ escrow.lex

> *The Filing Office may, if Default **is** declared, pay the Digital Asset Collateral to the Secured Party.*

see ⟶ statement.lex

Note that in the following example, License means the (sub)contract itself and **is** checks the state of the License, diverting into the clause Factually Breached to find out if the License is breached.

> *The Arbiter may, if this License **is** Factually Breached:*

see ⟶ evaluation.lex

itself

reflexive pronoun
`automatic reference`

Refers to the subject of the sentence.

Functions like ⟶ **themself**.

least

adjective
`time operator part`

Only in conjunction with ⟶ **at**.

> *"Factually Breached" is defined as: this License is Commissioned and the Comment Text is not fixed, or this License is Published and there is no Permission to Comment and the Notice Time lies at **least** 24 hours in the past.*

see ⟶ evaluation.lex

lex

noun
`keyword`

Keyword for the start of a digital contract.

LEX must be the first word of a digital contract. The words after **LEX** are the name of the entire digital contract (system) described thereafter.

As a convention, **LEX** is usually spelled in uppercase.

> ***LEX** Escrow.*

see ⟶ escrow.lex

lexon

noun
`keyword`

The numbers following **LEXON** are the **version number** of the Lexon compiler that the digital contract was written for. This is a concept that helps while Lexon is evolving. As a rule, newer compilers can compile older version Lexon texts but there will sometimes be 'breaking changes' where this backward compatibility is not provided and older texts have to be adapted to the changes of a new grammar.

Note that compatibility is not a dimension of what the texts mean in human language, which remains the same throughout Lexon versions, because English does not change. Instead, this is about older versions of the compiler understanding less than newer ones, i.e., the grammar getting less restricted.

As a convention, **LEXON** is usually spelled in uppercase.

> ***LEXON**: 0.2.12*

see ⟶ statement.lex

lies

verb
`time comparison operator`

Only in conjunction with ⟶ **at least**.

> *"Factually Breached" is defined as: this License is Commissioned and the Comment Text is not fixed, or this License is Published and there is no Permission to Comment and the Notice Time **lies** at least 24 hours in the past.*

see ⟶ evaluation.lex

may

auxilliary verb
`permission marker`

The subject to **may** is/are the only party or parties to the contract that are auhtorized to initiate the action described. The subject must be bound, i.e., the name before **may** must have been defined before, it cannot be defined in the may statement. Note that statements without may might likewise restrict authority to the named subject. And it is possible that the subject is unbound in cases without **may**, i.e., the role not defined at that point.

> *The Filing Office **may** certify the File Number.*

see ⟶ statement.lex

millisecond, milliseconds

noun
`time constant of 1/1000 seconds.`

Functions like ⟶ **days**.

minute, minutes

noun
`time constant of 60 seconds`

Functions like ⟶ **days**.

month, months

noun
`Time constant of 2592000 seconds, i.e., 30 DAYS`

Functions like ⟶ **days**.

myself

reflexive pronoun
`automatic reference`

Refers to the subject of the sentence.

Functions like ⟶ **themself**.

no

adverb
`logic operator`

Logical inversion. In conjunction with ⟶ **there is**, also used to test whether a name has been assign any concrete meaning yet. See ⟶ **fixed**.

> *"Factually Breached" is defined as: this License is Commissioned and the Comment Text is not fixed, or this License is Published and there is **no** Permission to Comment and the Notice Time lies at least 24 hours in the past.*

see ⟶ evaluation.lex

not

adverb
`logic operator`

Logical inversion.

Used to form the opposite of a logical expression. Can be positioned before a name, or before ⟶ **fixed**. The resulting expression means the opposite of what the part after not meant. It can be part of a bigger logical expression, as shown below. **Not**, as is grammatically correct, binds the next noun or verb only. The requirements for sentence structure make sure that no ambiguity can arise for the human reader.

If a more complex expression must be inverted, it has to be written as a clause. This simple device to avoid ambiguity without requiring literal bracketing is borrowed from proven best practice in Functional Programming.

> *"Factually Breached" is defined as: this License is Commissioned and the Comment Text is **not** fixed, or this License is Published and there is no Permission to Comment and the Notice Time lies at least 24 hours in the past.*

see ⟶ evaluation.lex

now

noun
`time value`

Synonym to ⟶ **current time**.

> *The Filing Office may fix the Initial Statement Date as **now**.*

of

preposition
`no op`

Only with ⟶ **remainder**.

> *The Arbiter may pay from escrow the Fee to themselves, and afterwards pay the remainder **of** the escrow to the Payee.*

see ⟶ escrow.lex

off

adverb
`operator part`

Only with ⟶ **signed**.

on

preposition
`operator part`

Only with ⟶ **signed**.

oneself

reflexive pronoun
`automatic reference`

Refers to the subject of the sentence.

Functions like ⟶ **themself**.

or

conjunction
`logic operator`

Used to build logical expressions. A phrase that contains **or** is ⟶ **true** if the part left or the part right of the **or** are true.

Colons, commas and semicolons are relevant to separate sub-phrases. Programmers note that there is no precedence of **and** over **or** in Lexon as this is not a part of natural language. Commas and semicolons offer two levels of nesting. Beyong this, precedence is created by encapsulating logical expressions into separate clauses.

In the example ...

> *"Factually Breached" is defined as: this License is Commissioned and the Comment Text is not fixed,* ***or*** *this License is Published and there is no Permission to Comment and the Notice Time lies at least 24 hours in the past.*

see ⟶ evaluation.lex

... all of the following counts as the left-side of the **or**, because there is a comma before the **or** and no comma to the left of it:

> *this License is Commissioned and the Comment Text is not fixed*

... and all of the following is the right side of the **or**, because there is a comma before the **or** and no comma to the right of it:

> *this License is Published and there is no Permission to Comment and the Notice Time lies at least 24 hours in the past.*

ourselves

reflexive pronoun
`automatic reference`

Refers to the subject of the sentence.

Functions like ⟶ **themself**.

passed

adjective
`time comparison operator`

Compares a point in time to the current time.

> *The Filing Office may, if the Continuation Window Start has **passed**, send the Notification Statement to the Secured Party.*

see ⟶ statement.lex

For more on how to use time, see ⟶ **hours** and ⟶ **days**.

past

noun or adjective
`negative time sign`

Past indicates that a measure of time is to be subtracted from the current time, or it functions like ⟶ **has passed**.

In the example, **in the past** functions as a negative sign to the literal 24 hours, relative to now.

> *"Factually Breached" is defined as: this License is Commissioned and the Comment Text is not fixed, or this License is Published and there is no Permission to Comment and the Notice Time lies at least 24 hours in the **past**.*

see ⟶ evaluation.lex

For more on how to use time, see ⟶ **hours** and ⟶ **days**.

pay, pays

verb
`transfer operator`

A transfer over the amount given immediately following **pay**, from the subject of the sentence, to the object marked with ⟶ **to** or ⟶ **into**.

> *The Payer **pays** an Amount into escrow, appoints the Payee, appoints the Arbiter, and fixes the Fee.*

see ⟶ escrow.lex

per

preposition
`keyword`

PER marks the beginning of the ⟶ **terms** of a covenant or sub-contract.

Digital contracts are often really contract systems that control the creation of individual contracts each with different counter parties. These sub contracts are called covenants in the context of digital contracts. Their terms are separated from the general terms of the digital contracts – which govern everything else, specifically how the covenants come into existence – by the keyword **per**, followed by the name the of the covenant, and optionally preceded by the keyword ⟶ **terms**.

As a convention, **PER** is usually spelled in uppercase.

> *TERMS **PER** License:*

see ⟶ statement.lex

person

noun
`type`

Defines a name to stand for a person.

> *"Payer" is a person.*

see ⟶ escrow.lex

preamble

noun
`keyword`

Start of a comment from both legal and processing point of view. Words after **PREAMBLE** are explanations with minimal legal weight and are not translated to 3rd generation language code by the compiler. Accordingly, in the example below, no word behind the colon is interpreted. This is not a special case: it is similar to how Lexon does not account for the common meaning of nouns in human language that a writer defines. This meaning is helpful to understand the contract, but not part of it, like the **PREAMBLE** text. Likewise, it is a common pitfall to read the preamble in a paper contract as part of the legal agreement; it is not. Its value lies in paraphrasing the more technical prose of the agreement in more accessible but blurier terms and to provide context.

As a convention, **PREAMBLE** is usually spelled in uppercase.

> ***PREAMBLE**: This is a licensing contract for a software evaluation.*

see ⟶ evaluation.lex

provided

adjective
`conditional keyword`

Synonym to ⟶ **if**.

> *The Filing Office may, **provided** the Continuation Window Start has passed, send the Notification Statement to the Secured Party.*

register, registers

verb
`parameter assignment operator`

Synonym to ⟶ **fix**.

> *The Licensee may **register** a Comment Text.*

see ⟶ evaluation.lex

remainder

noun
`no op`

Optional part to internal token count variable ⟶ **escrow**.

> *The Arbiter may pay from escrow the Fee to themselves, and afterwards pay the **remainder** of the escrow to the Payee.*

see ⟶ escrow.lex

respective

adjective
`optional part of built-in time value`

Only in conjunction with ⟶ **current time**.

> *The Secured Party may certify the Termination Statement Time as the **respective** current time.*

return, returns

verb
`transfer operator`

Synonym to ⟶ **pay**.

> *The Arbiter may pay from escrow the Fee to themselves, and afterwards **return** the remainder of the escrow to the Payer.*

see ⟶ escrow.lex

second, seconds

noun
`time constant of 1 second`

Functions like ⟶ **days**.

send, sends

verb
`transfer and messaging operator`

Send a message. On the blockchain, this can be an entry on the receipt log.

> *The Filing Office may, if the Continuation Window Start has passed, **send** the Notification Statement to the Secured Party.*

see ⟶ statement.lex

signed

adjective
`logical true`

Only in the combination **signed off**, with optional following ⟶ **on**.

In other words, **signed**, **signed off**, and **signed off on** all mean the same.

> *The Agent may, once the Receipt is **signed** off, return the Collateral to the Counterparty.*

so

adjective
`causal concatenator part`

Only in conjunction with ⟶ **if**.

if so is a synonym for ⟶ **afterwards**. See remarks on sentence order there.

> *The Arbiter may pay from escrow the Fee to themselves, and if **so** return the remainder of the escrow to the Payer.*

terminate, terminates

verb
`destruction operator`

The consequence of termination is that a contract's state cannot be changed anymore. Both main contracts and covenants (subcontracts) can be terminated. It is good practice to end a contract after its purpose is fulfilled so that it cannot be partially restarted for unintended consequences.

> *The Filing Office may, if the Termination Period has passed,* ***terminate*** *this contract.*

see ⟶ statement.lex

terms

noun
`optional keyword`

Optional marker of the beginning of general or per-subcontract terms. The **TERMS** keyword serves to increase clarity but can be left out as the document order suffices for the compiler to understand what part of a document to expect next: terms are neccessarily all statements following the ⟶ **LEX** keyword and digital contract (system) name. For yet more clarity, **TERMS** can be preceded by the optional keyword ⟶ **GENERAL**.

For covenants (sub contracts), their terms must be marked at least by the keyword ⟶ **PER**, followed by the covenant's name. **TERMS** may precede ⟶ **PER** but is optional.

As a convention, **TERMS** is usually spelled in uppercase.

> ***TERMS****:*

see ⟶ evaluation.lex

> ***TERMS*** *PER License:*

see ⟶ statement.lex

text

noun
`type`

Defines that a name is standing for a text.

> *"Notification Statement" is a **text**.*

see ⟶ statement.lex

that

conjunction
`conditional keyword`

Following statements are executed only of the immediately following condition is true.
Appears only together with ⟶ **given**.

given that is a synonym to ⟶ **if**.

> *The Filing Office may, given **that** the Continuation Window Start has passed, send the Notification Statement to the Secured Party.*

the

article
`no op`

Articles are optional to increase readability, because the name they precede must always be unambigous on its own.

> *The Payer pays an Amount into escrow, appoints **the** Payee, appoints **the** Arbiter, and fixes **the** Fee.*

see ⟶ escrow.lex

Cf. ⟶ **a**.

themself, themselves

reflexive pronoun
`automatic reference`

Refers to the subject of the sentence.

In this example, **themselves** means the Arbiter.

> *The **Arbiter** may pay from escrow the Fee to **themselves**, and afterwards pay the remainder of the escrow to the Payee.*

see ⟶ escrow.lex

then

adverb, adjective
`conditional keyword, causal conccatenator`

> *If the Termination Period has passed, **then** terminate this contract.*

In conjunction with ⟶ **current time**:

> *The Secured Party may file a Termination Statement, and certify the Termination Statement Time as the **then** current time.*

see ⟶ statement.lex

Also functions like ⟶ **therefor**:

> *This License is **then** Paid.*

there

adverb
`existence test`

Used to reason about the existence of something, ore more precisely, about whether a name has a defined meaning or not.

Appears only in **there is** or **there is not**, or with variations of ⟶ **is**, like ⟶ **be**.

Cf. ⟶ **fixed**.

> *"Factually Breached" is defined as: this License is Commissioned and the Comment Text is not fixed, or this License is Published and **there** is no Permission to Comment and the Notice Time lies at least 24 hours in the past.*

see ⟶ evaluation.lex

therefor, therefore

adverb
`causal concatenator`

Therefore binds a sentence to the preceding ones, so that it is performed only if all preceding sentences were performed, i.e., did not disqualify for access or conditional reasons.

Without **therefore**, a sentence by itself is always materialized when a clause is triggered.

Cf. ⟶ **afterwards**.

> *The Licensee pays the Licensing Fee to the Licensor, and pays the Breach Fee into escrow. This License is **therefore** Paid.*

see ⟶ evaluation.lex

these

adjective
`no op`

These can be required to get the natural language grammar right but does not change meaning by its presence or absence because the name that it precedes must always be unambiguous by itself.

> *The Licensor may certify **these** Agreements as Commissioned.*

this

adjective
`no op`

this can be required to get the natural language grammar right but does not change meaning by its presence or absence because the name that it precedes must always be unambiguous by itself.

> *The Licensor may certify **this** License as Commissioned.*

see ⟶ evaluation.lex

time

noun
`type`

Either defines a name as standing for a time value.

> *"Initial Statement Date" is a **time**.*

see ⟶ statement.lex

Or, specificies the ⟶ **current** point in time.

> *The Filing Office may fix the Initial Statement Date as the current **time**.*

see ⟶ statement.lex

to

preposition
`transfer operator part`

Appears in conjunction with ⟶ **pay**, ⟶ **send**, ⟶ **be** or ⟶ **equal**.

> *The Arbiter may pay from escrow the Fee **to** themselves, and afterwards pay the remainder of the escrow **to** the Payee.*

see ⟶ escrow.lex

> *The Filing Office may, if the Continuation Window Start has passed, send the Notification Statement **to** the Secured Party.*

see ⟶ statement.lex

true

adjective
`logical true`

A value that a name or an expression can have, meaning that something is the case.

Synonymous with ⟶ **yes**.

In the following example, the expression *the Continuation Window has passed* can be ⟶ **true** or ⟶ **false**.

> *The Filing Office may, if **the Continuation Window Start has passed**, send the Notification Statement to the Secured Party.*

see ⟶ statement.lex

Cf. ⟶ **fixed**.

was

verb
`logic equivalence operator`

Functions like ⟶ **is**.

week, weeks

noun
`time factor constant`

Time constant of 604,800 seconds, i.e., 7 ⟶ **days**.

Functions similar to ⟶ **days**.

with

conjunction
`causal concatenator`

Only appears as **and with this**.

Functions like ⟶ **therefore**.

year, years

noun
time factor constant

Time constant of 31,536,000 seconds, i.e., 365 ⟶ **days**.

Functions simlar to ⟶ **days**.

yes

noun
`logical true`

A value that a name or an expression can have, meaning that something is the case.

Synonymous with ⟶ **true**.

yourself, yourselves

reflexive pronoun
`automatic reference`

Refers to the subject of the sentence.

Functions like ⟶ **themself**.

Examples

Minimal Escrow Example

This is the example that frequently features in the Lexon material, from the ⟶ *book* to the ⟶ *whitepaper*.

LEX Escrow.

"Payer" is a person.
"Payee" is a person.
"Arbiter" is a person.
"Amount" is an amount.
"Fee" is an amount.

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

CLAUSE: Pay Out.
The Arbiter may pay from escrow the Fee to themselves, and afterwards pay the remainder of the escrow to the Payee.

CLAUSE: Pay Back.
The Arbiter may pay from escrow the Fee to themselves, and afterwards return the remainder of the escrow to the Payer.

Carla L. Reyes, U.C.C. Statement

This Lexon text is model trade statute developed by asst. prof. Carla L. Reyes. For an in-depth, abstract and concrete *legal* discussion see ⟶ *Reyes, Carla, Creating Cryptolaw for the Uniform Commercial Code (2021)*. 78 Washington and Lee Law Review 1521 (2021), SMU Dedman School of Law Legal Studies Research Paper No. 502. For additional technical details, including a terminal run-through, see ⟶ *lexon.org/reyes*.

LEX UCC Financing Statement.

LEXON: 0.2.12

"Financing Statement" is this contract.
"File Number" is data.
"Initial Statement Date" is a time.
"Filer" is a person.
"Debtor" is a person.
"Secured Party" is a person.
"Filing Office" is a person.
"Collateral" is data.
"Digital Asset Collateral" is an amount.
"Reminder Fee" is an amount.
"Continuation Window Start" is a time.
"Continuation Statement Date" is a time.
"Continuation Statement Filing Number" is data.
"Lapse Date" is a time.
"Default" is a binary.
"Continuation Statement" is a binary.
"Termination Statement" is a binary.
"Termination Statement Time" is a time.
"Notification Statement" is a text.

The Filer fixes the Filing Office, fixes the Debtor, fixes the Secured Party, and fixes the Collateral.

Clause: Certify.
The Filing Office may certify the File Number.

Clause: Set File Date.
The Filing Office may fix the Initial Statement Date as the current time.

Clause: Set Lapse.
The Filing Office may fix the Lapse Date.

Clause: Set Continuation Start.
The Filing Office may fix the Continuation Window Start.

Clause: Pay Fee.
The Secured Party may pay a Reminder Fee into escrow.

Clause: Notice.
The Filing Office may fix the Notification Statement.

Clause: Notify.
The Filing Office may, if the Continuation Window Start has passed, send the Notification Statement to the Secured Party.

Clause: Pay Escrow In.
The Debtor may pay the Digital Asset Collateral into escrow.

Clause: Fail to Pay.
The Secured Party may declare Default.

Clause: Take Possession.
The Filing Office may, if Default is declared, pay the Digital Asset Collateral to the Secured Party.

Clause: File Continuation.
The Secured Party may file the Continuation Statement.

Clause: Set Continuation Lapse.
The Filing Office may, if the Continuation Statement is filed, fix the Continuation Statement Date.

Clause: File Termination.
The Secured Party may file a Termination Statement, and certify the Termination Statement Time as the then current time.

Clause: Release Escrow.
The Filing Office may, if the Termination Statement is filed, return the Digital Asset Collateral to the Debtor.

Clause: Release Reminder Fee.
The Filing Office may, if the Termination Statement is filed, return the Reminder Fee to the Secured Party.

Clause: Termination Period.
"Termination Period" is defined as 365 days after the Termination Statement Time.

Clause: Terminate and Clear.
The Filing Office may, if the Termination Period has passed, terminate this contract.

Florian Idelberger, License Evaluation

This digital contract was created by Florian Idelberger, Phd candidate at the European University Institute in Florence. It appears in ⟶ *Merging traditional contracts (or law) and (smart) e-contracts – a novel approach*, comparing this text to smart contracts written in other languages.

This test case is a license contract to license a copy of a software or other specified work for use and evaluation, in exchange for a licensing fee. Furthermore, sublicensees can be specified. These grants and license are defined in article 1. The sublicense part was inspired by Surden's description of a licensing system where universities can automatically manage the licenses of their libraries and conclude more tailored licensing agreements. In article 2, it is defined that optionally, the licensee or sublicensee is commissioned to publish comments about the use of the product. This approves publication, but also requires it. In article 3, publishing of comments about the use and evaluation of the asset without approval by the licensor beforehand is prohibited. In case of unauthorized publication, the licensee has 24 hours to remove the published material. This improves the test case, as it requires use of external agents or data sources depending on the system, as otherwise there is no basis on which to automate or act. Additionally, the passing of time is tested with the time limit. Finally, in article 4, it is laid out that the license agreement terminates automatically upon breach of the licensing agreement, and that the licensee has to pay a fee in case of breach. The resulting license agreement is simplified for publication, but includes the most important features of license contracts.

LEX: Evaluation License System.

LEXON: 0.2.1
AUTHORS: FLORIAN IDELBERGER, HENNING DIEDRICH

PREAMBLE: This is a licensing contract for a software evaluation.

TERMS:

"Licensor" is a person.
"Arbiter" is a person.
"Licensing Fee" is an amount.
"Breach Fee" is an amount.

The Licensor appoints the Arbiter,
fixes the Licensing Fee,
and fixes the Breach Fee.

TERMS PER License:

"Description of Goods" is a text.
"Licensee" is a person.
"Paid" is a binary.
"Commissioned" is a binary.
"Comment Text" is a text.
"Published" is a binary.

"Permission to Comment" is a binary.
"Notice Time" is a time.
"License" is this contract.

The Licensor appoints the Licensee, and fixes the Description of Goods.

CLAUSE: Pay.
The Licensee pays the Licensing Fee to the Licensor,
and pays the Breach Fee into escrow.
This License is therefore Paid.

CLAUSE: Commission.
The Licensor may certify this License as Commissioned.

CLAUSE: Comment.
The Licensee may register a Comment Text.

CLAUSE: Publication.
The Licensee may certify this License as Published.

CLAUSE: Grant Permission to Comment.
The Licensee may grant the Permission to Comment.

CLAUSE: Declare Breach.
The Arbiter may, if this License is Factually Breached:
pay the Breach Fee to the Licensor,
and afterwards terminate this License.

CLAUSE: Factually Breached.
"Factually Breached" is defined as:
this License is Commissioned and the Comment Text is not fixed,
or this License is Published and there is no Permission to Comment and the Notice Time lies at least 24 hours in the past.

CLAUSE: Notice.
The Licensor or the Arbiter may fix the Notice Time as the respective current time.

CLAUSE: Noticed.
"Noticed" is defined as a Notice Time being fixed.