Variable Definition Language (VDL)

1 Overview

Variables Definition Language (VDL) is a proprietary technology developed by P & L Software Publishing Limited. It integrates seamlessly with Microsoft Word to facilitate the creation and maintenance of CSA Format Files (document templates) and the intelligent generation of statutory forms, registers, letters, management reports, and other documents within CSA. VDL3 is the latest generation of the technology.

To implement Variables Definition Language, you will need a Unicode-enabled version of Microsoft Word for creating CSA Format Files—typically Microsoft Word 2000 or any later version—together with the VDL Interpreter for runtime execution, which is embedded within CSA Expert.

The published specifications of VDL are organized into three (3) categories, namely:

(1) VDL Functions

(2) Directives

(3) Section Delimiters

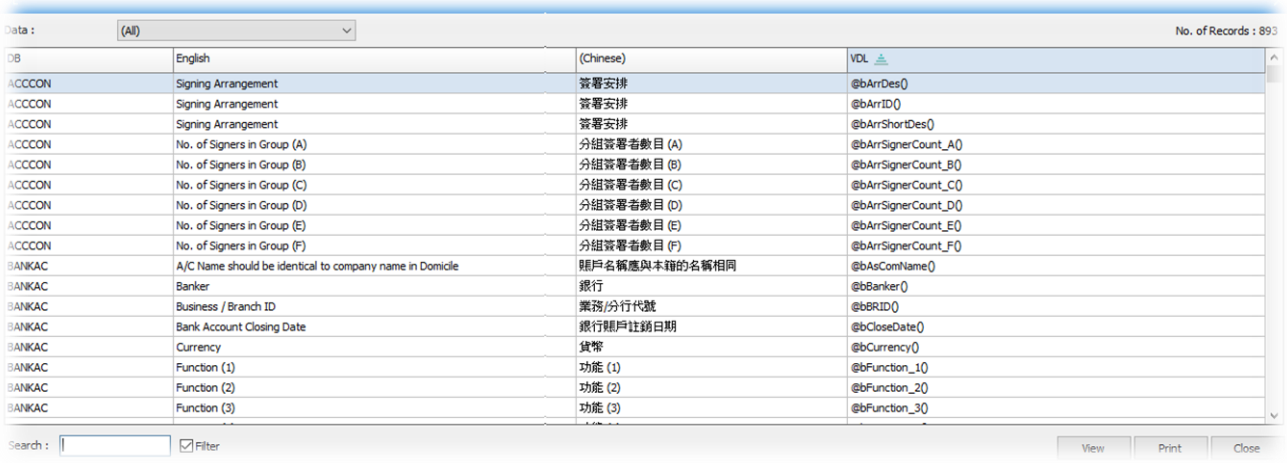
With these specifications, you can retrieve any data in CSA (including images) and present them in any appropriate way you want in any documents and reports.

Let's first take a look at the following examples:

Category	***Example***	***Explanation***
VDL Function	`@cAdd([<parameters>])`	This "Function" returns the value of the registered office address of a Client. Depending on the `<parameters>`, It can be the full address or any part of it.
Directive	`@SetFmtFont(<parameter>)`	This "Directive" causes the return values of all Functions within the whole document template to present in such font and such style as specified by the `<parameter>`.

Section Delimiters	`[Begin_English]`
	<contents>
	`[End_English]`
	
	`[Begin_Chinese]`
	<内容>
	`[End_Chinese]`
		The two pairs of Section Delimiters respond according to your "Document Language" setting during document generation.
		If the document is generated in English, the <contents> enclosed within the [Begin English] / [End English] delimiters will be preserved; the Chinese-specific section will be repelled.
		Conversely, if the document is generated in Chinese, the <内容> enclosed within the [Begin Chinese] / [End Chinese] delimiters will be preserved; the English-specific section will be repelled.

There are more than 900 VDL Functions available in CSA. For the complete list, navigate to { System > Document Template Library > VDL } within CSA. This library serves as an excellent reference when tailoring your own versions of document templates.



Data : (All) No. of Records : 893

DB	English	(Chinese)	VDL
ACCCON	Signing Arrangement	簽署安排	@bArrDes()
ACCCON	Signing Arrangement	簽署安排	@bArrID()
ACCCON	Signing Arrangement	簽署安排	@bArrShortDes()
ACCCON	No. of Signers in Group (A)	分組簽署者數目 (A)	@bArrSignerCount_A()
ACCCON	No. of Signers in Group (B)	分組簽署者數目 (B)	@bArrSignerCount_B()
ACCCON	No. of Signers in Group (C)	分組簽署者數目 (C)	@bArrSignerCount_C()
ACCCON	No. of Signers in Group (D)	分組簽署者數目 (D)	@bArrSignerCount_D()
ACCCON	No. of Signers in Group (E)	分組簽署者數目 (E)	@bArrSignerCount_E()
ACCCON	No. of Signers in Group (F)	分組簽署者數目 (F)	@bArrSignerCount_F()
BANKAC	A/C Name should be identical to company name in Domicile	賬戶名稱應與本籍的名稱相同	@bAsComName()
BANKAC	Banker	銀行	@bBanker()
BANKAC	Business / Branch ID	業務/分行代號	@bBRID()
BANKAC	Bank Account Closing Date	銀行賬戶註銷日期	@bCloseDate()
BANKAC	Currency	貨幣	@bCurrency()
BANKAC	Function (1)	功能 (1)	@bFunction_1()
BANKAC	Function (2)	功能 (2)	@bFunction_2()
BANKAC	Function (3)	功能 (3)	@bFunction_3()

Search : Filter View Print Close

2 VDL Functions

2.1 Introduction

VDL Function starts with an at-sign "@", follows by a Function Name, and ends with a pair of brackets "( )" with or without Function Parameters (roughly mean “settings”) inside the brackets.

Syntax: @<function_name>(<function_parameters>)

Each VDL Function returns a value (result), representing a piece or block of data. When you instruct CSA to generate a document, the values returned by functions will be interpreted into "real" information (that can be a company name, an address, a list of directors, or even an image).

Example: `@cAdd(Country)`
Return Value: United States of America

In the above example, the VDL Function returns the country of a company's registered office address by specifying a “Data Parameter” (i.e. Country) to the function. As a matter of fact, you can control the output effectively by using up to five (5) parameters.

@<function_name>(<Data>, <Substitution>, <Modifier>, <Function>, <Font>)

VDL Function Name

VDL Function Name is used to identify the VDL Function itself. To reduce redundancy, most VDL Function names and parameters use generally accepted short forms to represent their meanings. In `@cAdd()`, the small letter "c" stands for "Client" (data file descriptor), and "Add" stands for "Address" (content descriptor), which means: the registered office **Add**ress field in the **c**lient data table.

VDL Parameter

Parameters are used to specify which particular information to retrieve and in what manner it is to be presented in the output document. The following examples use the “Data parameter” to determine which part (component) of the name of an Entity to retrieve.

VDLFunction	*Data to Retrieve*	*Return Value*
`@mName(EngPrefix)`	English prefix	Mr.
`@mName(EngName1)`	English surname	CHAN
`@mName(EngName2)`	English forename	Tai Man
`@mName(EngFormal)`	English formal name	Mr. CHAN Tai Man

VDL Function Types and CSA Data Types

VDL function types represent CSA data types. There are altogether nine (9) VDL Function types, each of them corresponds with a CSA data type:

#	*Function Type*	*Example*
1	Address	Registered office address
2	Bilingual	Name of a class of shares (Ordinary share 普通股)
3	Date	Date of incorporation
4	Definition	Currency code
5	Entity Name	Legal name, former name, alias, etc.
6	Entity Identity	Passport no., company no. etc.
7	Graphical	Specimen signature of a signing officer of bank account
8	Logical	A register is (or is not) kept at the registered office
9	Numerical	No. of shares of a particular class held by a member

General Rules of Applying VDL Functions

Users should be aware of the basic rules for applying VDL Functions. Failure to follow these rules will cause the function to be treated as normal text, and no value as expected will be returned by the VDL Interpreter.

(1) Every VDL Function must start with an at-sign "@".

(2) Every VDL Function must end with a pair of brackets "( )", with or without parameter(s).

(3) Blank space is not allowed between characters within a VDL Function Name.

(4) If the VDL Function has parameters, they must be listed in the correct order and separated by commas.

2.2 VDL Function Parameters

In the above examples of Chan Tai Man's name, we have only illustrated the use of one type of VDL Function Parameter (i.e. the Data Parameter). As a matter of fact, you can control the results of output precisely by using up to five (5) parameters:

(1) Data Parameter

(2) Substitution Parameter

(3) Modifier Parameter

(4) Function Parameter

(5) Font Parameter

Parameters are optional. But if they are being specified, they must be in their designated positions and separated by commas:

@VDL_Name(<Data> , <Substitution> , <Modifier> , <Function> , <Font>)

In most cases, not all parameters are necessary since a default value will be given if a parameter is missing. Given that a company's incorporation date is 13/09/2018, the following examples illustrate how parameters work:

Example (1)	:	`@cIncorpDate(F,%DATE,^HYP|^HYP,Legal,`*Font*`|字體)`
Return Value	:	*-13th day of September, 2018-*
Explanation	:	It is like requesting CSA to "Return the full (`F`) incorporation date (`cIncorpDate`) in legal format (`Legal`), apply the specified font setting (*Font*|字體), and enclose the output with a pair of hyphens (`^HYP|^HYP`). In addition, if the date is empty, substitute the value with "[DATE]" (`%DATE`) to alert me."
Example (2)	:	`@cIncorpDate(F,%DATE,^AST|^AST,ChiClassic,`Font`|`*字體*`)`
Return Value	:	**二零一八年九月十三日**
Example (3)	:	`@cIncorpDate()`
Return Value	:	13/09/2018
Explanation	:	Even without any paramenter, the VDL Interpreter can still return the required date in its default manner.
Example (4)	:	`@cNextAnnDate(,,,IncDay#42)`
Return Value	:	25/10/2019
Explanation	:	With the fourth parameter—the Function Parameter—you can further manipulate the retrieved data to serve a specific purpose. In this example, the function adds 42 days to the next anniversary of the incorporation date, representing the statutory deadline for filing an annual return.

Any trailing comma that follows the last specified parameter is not compulsory. Therefore the following two structures are equally effective:

@<function_name>(<data_parameter>,,,,)

@<function_name>(<data_parameter>)

Note that in the above examples (1) and (2), both Data and Font parameters can be omitted if you simply want the full date and to have it displayed in default fonts, as in the case of examples (3) or (4).

#	*Parameter*	*Example*	*Meaning*
1	Data	`F`	“F” stands for Full date (the default parameter). Other options include D (day), M (month), Y (year), and W (day of week).
2	Substitution	`%DATE`	If the result is empty, it will be replaced with “[Date]” or “[日期]” (depending on the Document Language setting) for reminder purposes. CSA comes with a complete set of bilingual substitution codes for ready usage.
3	Modifier	`^HYP|^HYP`	Prefix and suffix a hyphen "–" respectively to the result.
4	Function	`IncDay#n`	This function Instructs the VDL Interpreter to add n days to the original date, and return the calculated result.
5	Font	Font|字體	Specify font attributes (type, size, and style) for the result. "Font" is for alpha-numerical characters and "字體" is for Chinese characters. Font settings can be simplified and automated by maintaining the font definition file `FONT.FMT` (that comes with CSA) and applying a pre-defined font control by using the `@SetFmtFont(parameter)` directive; or simply omitting the parameter for using default fonts.

2.3 The First Parameter: Data

Data Parameter (or the first parameter) specifies which related data you want to retrieve from the database. It is to be placed in front of the first comma:

Syntax: @<function_name>(<data_parameter>,,,,)

Below, we take the VDL Function of R/O address again as an example.

Example	*Return Value*
`@cAdd(EngLine1)`	8A, China Building,
`@cAdd(EngLine2)`	123 Broadway Path,
`@cAdd(EngLine3)`	Central, Hong Kong
`@cAdd(English)`	8A, China Building, 123 Broadway Path, Central, Hong Kong.

The Extended Parameter

Data Parameter has an extended parameter called "Alternate Data". It is to be placed next to the original Data Parameter, leading with a vertical line as the separator:

Syntax: @<function_name>(<data_parameter>|<alternate_data_parameter>,,,,)

The idea is: In case the original data (the return value of <data_parameter>) is empty, <alternate_data_parameter> can be used as a replacement. In other words, <data_parameter> is the first choice, and <alternate_data_parameter> is the second choice. It makes a lot of sense for most data types that are bilingual in nature.

Suppose a Client has only entered the English version of R/O address, using the Alternate Data Parameter ensures that the required data be able to present, whether in Chinese or English.

Example	***Return Value***
`@cAdd(English)`	8A, China Building, 123 Broadway Path, Central, Hong Kong.
`@cAdd(Chinese)`	(empty) / (空白)
`@cAdd(Chinese|English)`	8A, China Building, 123 Broadway Path, Central, Hong Kong.

Applicability

This parameter applies to all VDL Function types except the Graphical and the Numerical.

2.4 The Second Parameter: Substitution

Substitution Parameter (the second parameter) specifies what to substitute further in case both returned values of the Data Parameter and the Alternate Data Parameter are empty. It is to be placed between the first and second commas:

Syntax: @<function_name>(,<substitution_parameter>,,,)

The biggest advantage of using this parameter is to reduce the chance of data missing at the output level by providing a self-explanatory message in the form of return value. Another typical usage is to explicitly fill in something in a form to indicate that the returned value is empty (e.g. to automatically fill in "Nil" to the HKID Card No. column if an Entity is not a Hong Kong resident).

Suppose a Client only has an English version of R/O address, but you are required to fill in its Chinese version in a form. By applying the Substitution Parameter, you will be alerted by the returned value “[地址]” that there may have some data missing when you inspect the output document.

Example	***Return Value***
`@cAdd(English)`	8A, China Building, 123 Broadway Path, Central, Hong Kong.
`@cAdd(Chinese)`	(empty)
`@cAdd(Chinese,%ADD)`	[地址]

Relationship with the Data Parameter

The Substitution Parameter is closely related to the Data parameter. CSA will first look for <data_parameter> (the original parameter of the Data Parameter) and, if it is empty, then <alternate_data_parameter> (the extended parameter of the Data Parameter), and, if it is still empty, then use the Substitution Parameter.

List of Reserved Words

#	*Reserved Word*	*English Substitution*	*Chinese Substitution*
1	%DATE	[Date]	[日期]
2	%TIME	[Time]	[時間]
3	%MP	[Meeting Place]	[會議地點]
4	%MC	[Meeting Chairman]	[會議主席]
5	%NAME	[Name]	[名稱]
6	%ADD	[Address]	[地址]
7	%SHARE	[No. of Share]	[股份數目]
8	%CLASS	[Class of Share]	[股份類別]
9	%CURR	[Currency]	[貨幣]
10	%SIGNER	[Signer]	[簽署者]
11	%ST	[Signer Type]	[簽署者職銜]
12	%AMT	[Amount]	[款額]
13	%PC	[Percentage]	[百分比率]
14	%NA	N/A	不適用
15	%NIL	Nil	無

Applicability

This parameter applies to all VDL Function types.

2.5 The Third Parameter: Modifier

Modifier Parameter (the third parameter) puts additional characters or words before and / or after the returned value to make the context more complete or meaningful. It is placed between the second and third commas:

Syntax: @<function_name>(,,<modifier_parameter>,,)

The Two Segments of the Parameter

The Modifier Parameter is divided into two segments to represent the left-hand side and right-hand side contents respectively. The two segments <Prefix> and <Suffix> should be arranged in the following order, separated by a vertical line:

Syntax: @<function_name>(,,<prefix_modifier>|<suffix_modifier>,,)

If the <prefix_modifier> segment is specified, its content will be put on the left-hand side of the returned value. If the <suffix_modifier> segment is specified, its content will be put on the right-hand side of the returned value. Modifier Parameter will not cause changes to the returned value itself. It only adds additional content on any one side or both sides. If the returned value is empty, both <prefix_modifier> and <suffix_modifier> will then be ignored.

Example (1)	:	`@cName_Old()`
Return Value	:	XYZ Company Limited
Example (2)	:	`@cName_Old(,,^OB+Formerly+^SPA+known+^SPA+as+^SPA|^CB)`
Return Value	:	(Formerly known as XYZ Company Limited)
Example (3)	:	`@cName_Old(,,^OB+Formerly+^SPA+known+^SPA+as+^SPA|^CB)`
Return Value	:	(empty)
Explanation	:	Example (3) assumes that this Client does not have an “old” company name, thus both <prefix_modifier> and <suffix_modifier> are being ignored accordingly

Reserved Words and Operators

In Example (2), `^OB`, `^SPA` and `^CB` etc. are all reserved words. Each reserved word represents a non-alpha-numeric printable keyboard character or a Chinese punctuation mark, that is not allowed to use directly as part of the <prefix_modifier> or <suffix_modifier>. For example, "`@`", "`(`", and "`)`" will confuse the VDL Interpreter to identify the name or the parameters of the VDL Function. If you really mean to insert the "`@`", “(“, and “)” as modifiers, you should enter `^AT`, `^OB` (stand for **O**pen **B**racket), and `^CB` (stand for **C**lose **B**racket) respectively instead.

If you use more than one reserved words or use in conjunction with plain text, you must separate them with the "`+`" (plus sign) operator, as in Example (2).

Another designated operator for this parameter is the "`#`" (number sign) which is used to define the number of repetitions. For example, if you want to specify 3 consecutive hyphens (`---`) as the modifier, instead of writing `^HYP+^HYP+^HYP`, write `^HYP#3` as a shortcut.

Example	***Return Value***
`@gToday(,,^HYP+^HYP+^HYP+Date+^SPA|^HYP+^HYP+^HYP)`	`---Date 05/06/2007---`
`@gToday(,,^HYP#3+Date+^SPA|^HYP#3)`	`(same result)`

List of Reserved Words

Reserved Word	***Return Value***	***Meaning***
Brackets		
`^OAB`	<	Open Angle Bracket
`^CAB`	>	Close Angle Bracket
`^OCSAB`	〈	Open Chinese Single Angle Bracket
`^CCSAB`	〉	Close Chinese Single Angle Bracket
`^OCDAB`	《	Open Chinese Double Angle Bracket
`^CCDAB`	》	Close Chinese Double Angle Bracket
`^OB`	(	Open Bracket

Reserved Word	*Return Value*	*Meaning*
^CB	)	Close Bracket
^OCSCQM	「	Open Chinese Single Corner Quotation Mark
^CCSCQM	」	Close Chinese Single Corner Quotation Mark
^OCDCQM	『	Open Chinese Double Corner Quotation Mark
^CCDCQM	』	Open Chinese Double Corner Quotation Mark
^OCB	{	Open Curve Brace
^CCB	}	Close Curve Brace
^OSQM	‘	Open Single Quotation Mark
^CSQM	’	Close Single Quotation Mark
^OCSQM	「	Open Chinese Single Quotation Mark
^CCSQM	」	Close Chinese Single Quotation Mark
^ODQM	“	Open Double Quotation Mark
^CDQM	”	Close Double Quotation Mark
^OCDQM	『	Open Chinese Double Quotation Mark
^CCDQM	』	Close Chinese Double Quotation Mark
^OSB	[	Open Square Bracket
^CSB	]	Close Square Bracket
^OCSB	〔	Open Chinese Square Bracket
^CCSB	〕	Close Chinese Square Bracket
^OCHSB	【	Open Chinese Heavy Square Bracket
^CCHSB	】	Close Chinese Heavy Square Bracket
Symbols		
^AT	@	At
^AST	*	Asterisk
^BS	\	Backslash
^CAR	^	Caret
^COL	:	Colon
^CCOL	：	Chinese Colon
^COM	,	Comma
^CCOM	，	Chinese Comma
^DOL	$	Dollar
^EQU	=	Equal
^HYP	-	Hyphen
^NUM	#	Number
^PC	%	Percent
^PLUS	+	Plus
^SLA	/	Slash
^CSLA	／	Chinese Slash

Reserved Word	*Return Value*	*Meaning*
^SQM	'	Single Quotation Mark
^DQM	"	Double Quotation Mark
^EM	!	Exclamation Mark
^CEM	！	Chinese Exclamation Mark
^FS	.	Full Stop
^CFS	。	Chinese Full Stop
^QM	?	Question Mark
^CQM	？	Chinese Question Mark
^SC	;	Semicolon
^CSC	；	Chinese Semicolon
^SM	`	Stress Mark
^CLS	、	Chinese Listing Comma
^AMP	&	Ampersand
^CA	※	Chinese Ampersand
^TIL	~	Tilde
^CEM	…	Chinese Ellipsis Mark
^CPM	•	Chinese Partition Mark
^CHYP	—	Chinese Hyphen
^US	_	Underscore
^VL	|	Vertical Line
^NEWLINE		New Line
^SPA		Space
^TAB		Tab

Applicability

This parameter applies to all VDL Function types.

2.6 The Fourth Parameter: Function

Function Parameter (the fourth parameter) returns a converted result in accordance with its original value to fit in a specific purpose. It is placed between the third and last commas:

Syntax: @<function_name>(,,,<function_parameter>,)

In the first example below, all independent input fields of the R/O Address are joined together to form a full address as a single line. In the second example, input fields are outputted line-by-line according to the way they are entered in the Client Master File.

Different types of VDL Functions may have different reserved words for this parameter. Please refer to their respective sections for details.

Example	*Return Value*
`@cAdd()`	8A, China Building, 123 Broadway Path, Central, Hong Kong.
`@cAdd(,,,ByLine)`	8A, China Building, 123 Broadway Path, Central, Hong Kong.

Applicability

This parameter applies to all VDL Function types except the Entity Identity, the Graphical, and the Logical.

2.7 The Fifth Parameter: Font

Font Parameter (the fifth, or last, parameter) specifies the font type, size, and style of the value returned by a particular VDL Function. It is placed after the fourth, or last, comma:

Syntax: @<function_name>(,,,,<font_parameter>)

The Two Variants of the Parameter

The structure of this parameter can have two variants:

Variant	*Explanation*
`@VDL_Name(,,,,`Font|字體`)`	This structure divides the parameter into two segments, separated by a vertical line, representing the English font setting on the left-hand side and the Chinese font setting on the right-hand side.
`@VDL_Name(,,,,<FontID>)`	This structure accepts a pre-defined Font ID, stored in a modifiable font definition file `FONT.FMT`.

In normal situations, you do not need to use this parameter. CSA applies default font to present English and/or Chinese values returned by VDL Functions; so that, in general, all the returned values within the same document template will be of the same font type, size, and style. If for any reason you want a VDL Function applies a different font setting, you should use this parameter. To do so, either explicitly format the two parameter segments with MS-Word to your desired font settings (Variant 1) or simply assign an existing pre-defined Font ID (Variant 2) as the parameter.

In the first example below, CSA applies default fonts. In the second example, font settings as defined by the two parameter segments are used. In the third example, the settings of the Font ID `MyFont` as pre-defined in `FONT.FMT` (the font definition file) are used. Note that an identifier % (percentage sign) must be placed before the Font ID without space between them.

Example	**Return Value**
`@cAdd()`	香港中環百老匯徑 123 號 8A 室
`@cAdd(,,,,`Font|字體`)`	香港中環百老匯徑 123 號 8A 室
`@cAdd(,,,,%MyFont)`	**香港中環百老匯徑 123 號 8A 室**

Applicability

This parameter applies to all VDL Function types.

2.8 Address Type Functions

Information retrieved by an Address type VDL Function represents a universal address structure. This structure consists of two sets of records for English input and Chinese input respectively, each of which consists of three address lines, a Country field, and a Postal Code field. Every address can have its own presentation format. The default format is the full address in a single line.

Example (1)	:	`@cAdd()`
Return Value	:	<address_line_1> <address_line_2> <address_line_3> <country> <postal_code>
Example (2)	:	`@cAdd(,,,ByLine,)`
Return Value	:	<address_line_1> <address_line_2> <address_line_3> <country> <postal_code>

List of Reserved Words

As Data Parameter

Reserved Word	***Meaning***
`DefLang`	Retrieve the English or Chinese full address in a user-defined structure * (default reserved word)
`English`	Retrieve the English full address in a user-defined structure
`Chinese`	Retrieve the full Chinese address in a user-defined structure
`EngLine1`	English Address Line 1
`EngLine2`	English Address Line 2
`EngLine3`	English Address Line 3
`ChiLine1`	Chinese Address Line 1
`ChiLine2`	Chinese Address Line 2
`ChiLine3`	Chinese Address Line 3
`Country`	Retrieve the English or Chinese Country/Region field *
`EngCountry`	Country/Region in English
`ChiCountry`	Country/Region in Chinese
`PostCode`	Postal Code field (usually for non-Hong Kong addresses)
`EngFormat`	The address format setting for English output
`ChiFormat`	The address format setting for Chinese output

* As per the “Document Language” setting.

As Function Parameter

Reserved Word	*Meaning*	*Remark*
`ByLine`	List the full address line-by-line according to the input fields order or user's setting (i.e. Address Format).	Please see Example (1) & (2) above.

2.9 Bilingual Type Functions

Information to be retrieved by a Bilingual VDL Function represents a side-by-side bilingual data structure in English and Chinese (e.g. class of shares' name).

List of Reserved Words

As Data Parameter

Reserved Word	*Meaning*
`DefLang`	Retrieve either the English or Chinese field *
`English`	English field only
`Chinese`	Chinese field only
`EngChi`	Both English and Chinese fields, and in that order.
`ChiEng`	Both Chinese and English fields, and in that order.

* As per the “Document Language” setting.

As Function Parameter

Reserved Word	*Meaning*	*Example*

`ByLine`	Applicable only to `EngChi` and `ChiEng` Data Parameters.
	
	If `ByLine` is used, the two fields (or sets of fields) will be listed by lines, otherwise, they will be combined to form a single line with a space in between.
		`@VDL_Name(EngChi)`
		<English> <Chinese>
		
		`@VDL_Name(EngChi,,,ByLine)`
		<English>
		<Chinese>
		
		`@VDL_Name(ChiEng)`
		<Chinese> <English>
		
		`@VDL_Name(ChiEng,,,ByLine)`
		<Chinese>
		<English>

Address data type, Definition data type, and Entity Name data type are also bilingual in nature; but their unique structures stand out themselves as independent data types with their own VDL Functions.

2.10 Date Type Functions

Information to be retrieved by a Date type VDL Function represents a calendar date value.

List of Reserved Words

As Data Parameter

Data Parameter determines the content of a date to be retrieved. It can be a full date or one of its components (day, month, year, or day of week).

Reserved Word	*Meaning*	*Example*
`F`	Full Date	`@cIncDate(F)`
`D`	Day	`@cIncDate(D)`
`M`	Month	`@cIncDate(M)`
`Y`	Year	`@cIncDate(Y)`
`W`	Day of Week	`@cIncDate(W)`

As Function Parameter (for Presentation)

Function Parameters of Date type VDL Functions can be classified into two broad categories namely, **Presentation** Parameters and **Calculation** Parameters. The former determines how the date content is presented (including the format and the language); whereas the latter converts the original return date by increasing or decreasing its value (in terms of day, month or year) to derive a new date.

Reserved Word	*Meaning*	*Corresponding Data Parameter*	*Example*
For Presentation of Full Date			
`Short`	In short format (default reserved word)	`F`	13/06/2018 (as per My System Settings)
`DefLang`	Either in English or Chinese *	`F`	13th June, 2018
`English`	In English	`F`	13th June, 2018
`Chinese`	In Chinese	`F`	二零一八年六月十三日
`UK`	In British style	`F`	13th June, 2018
`UKShort`	In British style short format	`F`	13 Jun 2018
`Universal`	In Universal style	`F`	13 June 2018
`Legal`	In British style legal format	`F`	13th day of June, 2018
`UKDig`	In British style digital format with leading zero for day and month (i.e. dd/mm/yyyy)	`F`	13/06/2018
`UKShortDig`	In British style digital format *without* leading zero for day and month (i.e. d/m/yyyy)	`F`	13/6/2018
`US`	In American style	`F`	June 13, 2018

Reserved Word	*Meaning*	*Corresponding Data Parameter*	*Example*
`USShort`	In American style short format	`F`	Jun 13, 2018
`USShortDig`	In American style digital format *without* leading zero for month and day (i.e. m/d/yyyy)	`F`	6/13/2018
For Presentation of Day, Month, Year and Day of Week			

`DefLang`	In English or in Chinese * (default reserved word)	`D`
		`M`
		`Y`
		`W`
			13th
			June
			2018
			Wednesday

`English`	In English	`D`
		`M`
		`Y`
		`W`
			13th
			June
			2018
			Wednesday

`Chinese`	In Chinese	`D`
		`M`
		`Y`
		`W`
			十三
			六
			二零一八
			星期三

`ShortEng`	Month or Day of Week in English short form	`M` `W`	Jun (for June) Wed (for Wednesday)
`ShortChi`	Year or Day of Week in Chinese short form	`Y` `W`	一八 (for 二零一八) 三 (for 星期三)

`Dig`	Day or Month in digital format with leading zero, or Year in digital format with century	`D`
		`M`
		`Y`
			13
			06
			2018

`ShortDig`	Day or Month in digital format *without* leading zero, or Year in digital format *without* century	`D`
		`M`
		`Y`
			13
			6
			18

* As per “Document Language” setting.

As Function Parameter (for Calculation)

The basic syntax of a Calculation Parameter is `Inc<x>` or `Dec<x>`, where `<x>` can be `Day`, `Month` or `Year`, which means to increase or decrease one day, one month or one year. However, you can further use the special operator `"#"` (number sign) to specify your desired number of days, months, or years for increment or decrement; therefore `IncDay#10` increases by ten days, and `DecDay#10` decreases by ten days. The number assigned to the operator (i.e. 10 in the above example) must equal to or greater than 1.

It is possible to apply more than one Calculation Parameters. It is also possible to combine Presentation Parameter with Calculation Parameter(s). You do so by joining them with the `"+"` (plus sign) operator to form a single Function Parameter. For example, `UK+IncMonth+IncDay#2` increases one month and two days to the original date and presents it as a new date in British style.

Assuming that today is 5 June 2018 (05/06/2018):

Reserved Word	*Meaning*	*Example*
IncDay	Increase one day	@gDate(,,,IncDay) 6 June 2018
IncDay#n	Increase n days	@gDate(,,,IncDay#10) 15 July 2018
IncMonth	Increase one month	@gDate(,,,IncMonth) 5 July 2018
IncMonth#n	Increase n months	@gDate(,,,IncMonth#10) 5 April 2019
IncYear	Increase one year	@gDate(,,,IncYear) 5 June 2019
IncYear#n	Increase n years	@gDate(,,,IncYear#10) 5 June 2028
DecDay	Decrease one day	@gDate(,,,DecDay) 4 June 2018
DecDay#n	Decrease n days	@gDate(,,,DecDay#10) 26 May 2018
DecMonth	Decrease one month	@gDate(,,,DecMonth) 5 May 2018
DecMonth#n	Decrease n months	@gDate(,,,DecMonth#10) 5 August 2017
DecYear	Decrease one year	@gDate(,,,DecYear) 5 June 2017
DecYear#n	Decrease n years	@gDate(,,,DecYear#10) 5 June 2007
+	combine more than one function parameters	@gDate(,,,IncYear+IncDay,) 6 June, 2018 (increase 1 year and increase 1 day)

2.11 Definition Type Functions

Information retrieved by a Definition type VDL Function represents a record structure for most pre-defined codes (e.g. currency code). It usually consists of a Code field, an English Description field, a Chinese Description field, and a short Chinese Description field.

List of Reserved Words

As Data Parameter

Reserved Word	*Meaning*
Code	The Code (default reserved word)
Des	English or Chinese Description *
EngDes	English Description only
ChiDes	Chinese Description only
EngChiDes	Both English and Chinese Descriptions, and in that order.

ChiEngDes	Both Chinese and Chinese Description, and in that order.
ShortChiDes	Short Chinese Description
ShortDes	Either the Code (as if it is the English short description) or the Short Chinese Description *

* As per “Document Language” setting.

As Function Parameter

Reserved Word	*Meaning*	*Example*
ByLine	Applicable only to EngChi and ChiEng Data Parameters. If ByLine is used, the two fields (or sets of fields) will be listed by lines, otherwise, they will be combined to form a single line with a space in between.	@cJurisdiction(EngChiDes) Hong Kong SAR 香港特別行政區 @cJurisdiction(EngChiDes,,,ByLine) Hong Kong SAR 香港特別行政區 @cJurisdiction(ChiEngDes) 香港特別行政區 Hong Kong SAR @cJurisdiction(ChiEngDes,,,ByLine) 香港特別行政區 Hong Kong SAR

2.12 Entity Name Type Functions

Information retrieved by an Entity Name type VDL Function represents a universal name structure, which consists of all possible components related to the name of an individual Entity as well as a corporate Entity.

List of Reserved Words

As Data Parameter

Not all reserved words apply to all Entity types; some may only apply to Individual Entities. For “Entity Type” below “I” stands for Individual Entity, “C” stands for Corporate and all other non-Individual types of Entity.

Reserved Word	*Entity Type*	*Meaning*
Legal	I, C	Individual: Legal name in English or in Chinese * Corporate: Company name in English or in Chinese * (default reserved word)
Formal	I, C	Individual: Formal name in English or in Chinese * Corporate: Company name in English or in Chinese *
EngChi	I, C	Individual: Both English and Chinese legal names, and in that order. Corporate: Both English and Chinese company names, and in that order.

Reserved Word	*Entity Type*	*Meaning*
ChiEng	I, C	Individual: Both Chinese and English legal names, and in that order. Corporate: Both Chinese and English company names, and in that order.
EngChiFormal	I, C	Individual: Both English and Chinese formal names, and in that order. Corporate: Both English and Chinese company names, and in that order.
ChiEngFormal	I, C	Individual: Both Chinese and English formal names, and in that order. Corporate: Both Chinese and English company names, and in that order.
EngName1	I, C	Individual: English Surname Corporate: First line of English company name
EngName2	I, C	Individual: English Forename Corporate: Second line of English company name
ChiName1	I, C	Individual: Chinese Surname Corporate: First line of Chinese company name
ChiName2	I, C	Individual: Chines Forename Corporate: Second line of Chinese company name
English	I, C	Individual: English legal name Corporate: English company name
EngFormal	I, C	Individual: English formal name Corporate: English company name
Chinese	I, C	Individual: Chinese legal name Corporate: Chinese company name
ChiFormal	I, C	Individual: Chinese formal name Corporate: Chinese company name
Prefix	I	Prefix of an Individual in English or in Chinese *
Honour	I	Honour of an Individual in English or in Chinese *
Alias	I	Alias of an Individual in English or in Chinese *
OtherName	I	Other Name for an Individual
EngPrefix	I	Prefix of an Individual in English (e.g. Mr., Miss etc.)
EngHonour	I	Honour of an Individual in English (e.g. JP, Ph.D.)
EngAlias	I	English Alias of an Individual
EngLegalFormat	I	The English legal name format of an Individual
EngFormalFormat	I	The English formal name format of an Individual
ChiPrefix	I	Prefix of an Individual in Chinese (e.g. 先生、小姐)
ChiHonour	I	Honour of an Individual in Chinese (e.g. 太平紳士、博士)
ChiAlias	I	Chinese Alias of an Individual
ChiLegalFormat	I	The Chinese legal name format of an Individual
ChiFormalFormat	I	The Chinese formal name format of an Individual

* As per “Document Language” setting.

As Function Parameter

Reserved Word	*Meaning*	*Example*

`ByLine`	Applicable only to `EngChi` and `ChiEng` Data Parameters.
	
	If `ByLine` is used, the two legal names will be listed by lines, otherwise, they will be combined to form a single line with a space in between.
		`@mName(EngChi)`
		CHAN Tai Man 陳大文
		
		`@mName(EngChi,,,ByLine)`
		CHAN Tai Man
		陳大文
		
		`@mName(ChiEng)`
		陳大文 CHAN Tai Man
		
		`@mName(ChiEng,,,ByLine)`
		陳大文
		CHAN Tai Man

2.13 Entity Identity Type Functions

Information retrieved by an Entity Identity type VDL Function represents a data structure used to identify an Individual Entity, such as an ID card, passport, etc.

List of Reserved Words

As Data Parameter

Reserved Word	*Meaning*	*Example*
`ID`	Identity card number (default reserved word)	A123456(7)
`PP`	Passport number	K12345678
`IDType`	Name of the identity document in English or in Chinese *	
`EngIDType`	Name of the identity document in English	I.D. Card
`ChiIDType`	Name of the identity document in Chinese	身份證
`Country`	Issuing country of the identity document in English or in Chinese *	
`EngCountry`	Issuing country/region of the identity document in English	Hong Kong
`ChiCountry`	Issuing country/region of the identity document in Chinese	香港
`CountryCode`	Jurisdiction ID of the issuing country/region of the identity document	HKG
`IssDate`	Issue date of the identity document	01/12/2005
`ExpDate`	Expiry date of the identity document	01/12/2020

`Full`	All identity documents of an Entity, including Identity card, passport, and other identities, and in that order, line by line. * For example: China I.D. Card 30102019770707999X U.S.A. Passport 123456789 Singapore I.D. Card S7654321H	

* As per the “Document Language” setting.

As Function Parameter

`@mIdentity()` is the only VDL Function of this type that may have a Function Parameter. The reserved word is actually a Jurisdiction ID (`XXX`) and must be used along with `ID` or `PP` as the Data Parameter. It returns the number of an ID card or passport issued by the referred Jurisdiction ID as the issuing country or region. The following examples illustrate how it works:

VDL Function / Return Value	***Meaning***
`@mIdentity(ID,,PRC+^SPA+ID+^COL+^SPA,PRC,)` PRC ID: 30102019770707999X	China ID no. with a description specified in the Modifier Parameter.
`@mIdentity(PP,,HK+^SPA+Passport+^COL+^SPA,HKG,)` HK Passport: K12345678	Hong Kong passport no. with a description specified in the Modifier Parameter.

2.14 Graphical Type Functions

Information retrieved by a Graphical type VDL Function represents a graphical image. CSA database supports both BMP (Bitmap) and JPEG (Joint Photographic Experts Group) formats.

This type of VDL Function does not have any Data or Function parameter.

2.15 Logical Type Functions

Information retrieved by a Logical type VDL Function represents a logical value (also known as Boolean value), which is used to perform a True or False comparison on an expression.

Logical Functions make use of the Data Parameter to specify how the returned logical value is to be expressed. The original <data_parameter> parameter is used to specify the <True> side of the value, and the <alternate_data_parameter> parameter is used to specify the <False> side of the value. The two segments should be arranged in the following order, separated by a vertical line:

Syntax: @<function_name>(<True>|<False>,,,,)

Example	*Return Value*		*Remark*
	if True	*if False*	
`@cActive(Yes|No)`	Yes	No	
`@cActive(Yes)`	Yes		<False> segment not specified
`@cActive(Tick|Cross)`	✓	✗	
`@cActive(|Cross)`		✗	<True> segment not specified

List of Reserved Words

As Data Parameter

Reserved Word	*Return Value*	*Remark*
`Yes`	Yes or 是 * (default reserved word)	Text
`No`	No or 否	Text
`EngYes`	Yes	Text
`EngNo`	No	Text
`ChiYes`	是	Text
`ChiNo`	否	Text
`Tick`	✓	Symbol
`Cross`	✗	Symbol
`TickBox`	☑	Symbol
`CrossBox`	☒	Symbol
`EmptyBox`	☐	Symbol

* As per the “Document Language” setting.

Logical type VDL Function does not have a Function parameter.

2.16 Numerical Type Functions

Information retrieved by a Numerical type VDL Function represents a numerical value, which can be an integer or a real number (i.e. with a decimal point). The numerical value can be expressed in pure digital format (with or without thousand separators) or words (e.g. one million).

Numerical type of VDL Function does not have a Data parameter.

List of Reserved Words

As Function Parameter

Reserved Word	*Meaning*
`DefLang`	Express either in English or in modern Chinese words * (default reserved word)

Reserved Word	*Meaning*
`English`	Express in English words (e.g. "Ten Million" for 10,000,000)
`Chinese`	Express in modern Chinese words (e.g. 一千萬 for 10,000,000)
`ChiClassic`	Express in classical Chinese words (e.g. 壹千萬 for 10,000,000)
`NoThouSep`	Remove all thousand separators (e.g. 10000000 for 10,000,000)
`BlankZero`	Return blank for zero value
`InThousand`	Move decimal point left 3 places (e.g. 10,000 for 10,000,000)
`InTenThousand`	Move decimal point left 4 places (e.g. 1,000 for 10,000,000)
`InHundredThousand`	Move decimal point left 5 places (e.g. 100 for 10,000,000)
`InMillion`	Move decimal point left 6 places (e.g. 10 for 10,000,000)
`InTenMillion`	Move decimal point left 7 places (e.g. 1 for 10,000,000)
`InHundredMillion`	Move decimal point left 8 places (e.g. 0.1 for 10,000,000)
`InBillion`	Move decimal point left 9 places (e.g. 0.01 for 10,000,000)

* As per the “Document Language” setting.

Sometimes you may wish to express numerical values in "thousands" or other units instead of in their original formats. This is what exactly the `InThousand` and other similar Function parameters trying to do. For instance, if the issued share capital of a company is HK$10,000,000 and you want to express it in millions, you can apply the Function Parameter `InMillion` as follow:

Text : The Company's issued capital is HK$`@cCapIssued(,,,InMillion)` M.

Result : The Company's issued capital is HK$10M.

If you want to express the figure in English words, try to do this:

Text : The Company's issued capital is `@cCapIssued(,,,English)` Hong Kong Dollars.

Result : The Company's issued capital is Ten Million Hong Kong Dollars.

If you want to express the figure in Chinese words, try to do this:

Text : 本公司的已發行股本為港幣`@cCapIssued(,,,Chinese)`元。

Result : 本公司的已發行股本為港幣一千萬元。

3 VDL Directives

VDL Directive is the second element of the Variable Definition Language. Some VDL Directives are used to eliminate duplication and simplify complicated scrip, and some are used to produce certain special effects or formats in documents. Although the syntax of a VDL Directive is very much like that of a VDL Function, their meaning and usage are totally different. Simply speaking, a VDL Function returns information, whereas a VDL Directive issues an instruction to enhance the document generation mechanism. Some VDL Directives may have a parameter.

3..1 @SetFmtFont(<parameter>)

Other than using the default font control, you may instruct CSA to apply a pre-defined font control stored in the font definition file (i.e. FONT.FMT) to a particular document template by using the @SetFmtFont(<parameter>) directive. Its <parameter> is known as the “font control name”. That is to set font at Document template level instead of VDL Function level.

Given that you have defined a font control called "MyFont" in FONT.FMT:

```
@DocFont_MyFont(,,,,Font|字體)
```

By placing the VDL Directive @SetFmtFont(MyFont) in a document template (usually at its document header), returned values of all VDL Functions, except those with their own font settings, will follow the font type, size and style of @DocFont_MyFont(,,,,Font|字體).

3..2 @SetVDLGroup(<parameter>)

As implied by its name (Set VDL Group), @SetVDLGroup(<parameter>) provides a shorthand for you to refer to more than one VDL Functions at a time. The purpose is to define a “block” of VDL Functions in a single definition file, VDLGROUP.FMT, then apply it to several document templates. In case there are any changes in the contents of this block of VDL Functions, you just need to revise the definition file once rather than amend all the affected document templates one by one.

@SetVDLGroup(<parameter>) should be widely used to promote effectiveness and efficiency. Typically, it applied to the presentor's information, company name and company number in statutory forms, and the list of present directors at meeting minutes, just to name a few.

Example :

Given that you want to add a footnote, with User ID, Job No. and Our Ref., to all covering letters. Without this VDL Group function, you need to insert 3 VDL Functions to every relevant document template and set the font control for each VDL Function individually. It is absolutely a tedious business. To make matters worse, you need to do the entire task all over again in case of any change should be made to the footnote. The @SetVDLGroup(<parameter>) directive makes life easier by "doing once for all", thus eliminating repetition and simplifying your work.

First, create a new pair of Section Delimiters, give it a meaningful and easy-to-remember name, say, [Begin_LetterFootNote] … [End_LetterFootNote] in the definition file VDLGROUP.FMT, and insert the required VDL Functions inside them as follows:

```
[Begin_LetterFootNote]
@uID(,,,,%FootnoteFont)
@jNo(,,,,%FootnoteFont)
@jOurRef(,,,,%FootnoteFont)
[End_LetterFootNote]
```

Then, in the relevant covering letters, insert the following where it should be:

```
@SetVDLGroup(LetterFootNote)
```

Please note that @SetVDLGroup(<parameter>) processes only VDL Functions. All other things within the Section Delimiters will be ignored.

3..3 @SetRptAltShade()

Tables are widely used in reports, especially in checklists. In order to enhance the visual appeal of tabulated reports as well as clearly separate one band of data from another, CSA shades alternate rows the way as defined in the “sample data row”. `@SetRptAltShade()` is used to instruct the system to add this effect to the report.

Example：

Client ID	Company Name
[Begin_Report_Data_Client]	
@SetAltShade()	
@cID()	@cName()
[End_Report_Data_Client]	

Result：

Client ID	Company Name
ABC	ABC Company Limited
EFG	EFG Company Limited
TUV	TUV Company Limited
XYZ	XYZ Company Limited

3..4 @SetRowNo()

This directive is used to assign a table row number for a tabulated report.

Example：

No.	Client ID	Company Name
[Begin_Report_Data_Client]		
@SetAltShade()		
@SetRowNo()	@cID()	@cName()
[End_Report_Data_Client]		

Result：

No.	Client ID	Company Name
1	ABC	ABC Company Limited
2	EFG	EFG Company Limited
3	TUV	TUV Company Limited
4	XYZ	XYZ Company Limited

The true "beauty" of directives lies in the remarkable productivity they boost. The `@SetFmtBody(<parameter>)` and the other two directives greatly eliminate duplication of common scrip and simplify the complexity of maintaining document templates. Imagine that you have 50 document templates for meeting minutes, each has headings containing the company no., company name and its place of incorporation, etc. which are actually uniform for all files. If you have to modify something, you've got to do this 49 times even if you use the "cut and paste" method! This directive provides a shorthand for you to append the body of another document template (source file) to the calling document template (target file) where the directive is given.

3..5 @SetFmtHeader(<parameter>)

This directive provides a shorthand for you to append the page header of another document template to the calling document template where the directive is given. `<parameter>` refers to the name of the source document template.

3..6 @SetFmtBody(<parameter>)

This directive provides a shorthand for you to append the body of another document template to the calling document template where the directive is given. `<parameter>` refers to the name of the source document template.

3..7 @SetFmtFooter(<parameter>)

This directive provides a shorthand for you to append the page footer of another document template to the calling document template where the directive is given. `<parameter>` refers to the name of the source document template.

These three special directives work like the `@SetVDLGroup(<parameter>)` directive, but the <parameter> referred to is a file and not a section within a file. You can give the source file any valid name as long as it is saved as a valid RTF (Rich Text Format) file readable by Microsoft Word.

Given that a source document template name is `_MinHeader.MIE` (means Minutes Header):

Example：

Source File：

Company No. : @cComNo()
@cName(EngChi,,,ByLine)
(Incorporated in @cJurisdiction(Des))
(the "Company")

Target File：

@SetFmtBody(_MinHeader.MIE)

MINUTES OF MEETING OF THE BOARD OF DIRECTORS of the Company held at …

Result：

Company No. : 123456

Hong Kong International Company Limited
香港國際有限公司
(Incorporated in Hong Kong)
(the "Company")

MINUTES OF MEETING OF THE BOARD OF DIRECTORS of the Company held at …

The analogy is like computer programming. The main routine calls a sub-routine, and the sub-routine returns the expected results to the main routine. Simple but powerful, it saves time and avoids unnecessary errors.

4. VDL Section Delimiters

Section Delimiters is a pair of formatted phrases, used as indicators to specify the opening and closing of a section in a document template. They are widely used in minutes, resolutions, and reports.

4..1 Section Delimiters in Minutes

The document generation mechanism for minutes of meetings involves not only the insertion of information, but also the intelligence of retaining necessary sections or repealing undesired ones from the document templates.

For example, in the minutes of an Annual General Meeting, there is a paragraph stating whether the audited statement of accounts is available or not. Most people will have paragraphs of both situations pre-written in their AGM minutes template, and then, upon using the template, manually delete the inappropriate one in the final output document. Considering the AGM minutes may have many other similar cases, doing this manually for each case for each company is obviously error-proneness. CSA resolves this problem elegantly by using VDL Section Delimiters to retain or repeal sections automatically according to a specific criterion without user's interference.

The following example shows you how VDL Section Delimiters respond to cases in an AGM document template. Depending on the availability of the audited accounts, the appropriate or inappropriate section retains or repeals automatically.

Example：

DIRECTORS' REPORT AND AUDITED ACCOUNTS

[Begin_Accounts_Available]
RESOLVED THAT the audited statement of accounts together with the reports of the directors and auditors of the Company for the year ended @AcClosingDate(,%DATE,,English,) presented to shareholders by the Board be and are hereby received and adopted.
[End_Accounts_Available]
[Begin_Accounts_Not_Available]
NOTED THAT audited statement of accounts and reports of directors and auditors of the Company for the year ended @AcClosingDate(,%DATE,,English,) are not available for presentation to shareholders.
[End_Accounts_Not_Available]

Result (If accounts available)：

DIRECTORS' REPORT AND AUDITED ACCOUNTS

RESOLVED THAT the audited statement of accounts together with the reports of the directors and auditors of the Company for the year ended 31st March, 2019 presented to shareholders by the Board be and are hereby received and adopted.

Result (If accounts not available)：

DIRECTORS' REPORT AND AUDITED ACCOUNTS

NOTED THAT audited statement of accounts and reports of directors and auditors of the Company for the year ended 31st March, 2019 are not available for presentation to shareholders.

4..2 Section Delimiters in Reports

The meaning of Section Delimiters in Report is different from those of other types of document templates. They are not for selecting an appropriate section to retain or repeal, but to delimit a working area for a particular data set. There are three basic pairs of section delimiters for use in reports:

Report Section	***Section Delimiters***
Header	[Begin_Report_Header_<Data_Set_Keyword>] … [End_Report_Header_<Data_Set_Keyword>]
Data (or Body)	[Begin_Report_Data_<Data_Set_Keyword>] … [End_Report_Data_<Data_Set_Keyword>]
Footer	[Begin_Report_Footer_<Data_Set_Keyword>] … [End_Report_Footer_<Data_Set_Keyword>]

Example：

The following example shows you how they are used in reports to specify a dataset keyword, fix boundaries, and retrieve data from that specified dataset. Given that the keyword is GovComC (for Corporate Governance Committee):

[Begin_Report_Header_GovComC]

Committee ID	Committee Name

[End_Report_Header_GovComC]
[Begin_Report_Data_GovComC]
@SetAltShade()

@cGovCommittee()	@cGovCommittee(Des)

[End_Report_Data_GovComC]

Result：

Committee ID	Committee Name
AUDCOM	Audit Committee 審計委員會
CTCCOM	Connected Transactions Control Committee 關聯交易控制委員會
EXECOM	Executive Committee 執行委員會
REMCOM	Personnel and Remuneration Committee 人事和薪酬委員會

4..3 Section Delimiters and Document Language

VDL Section Delimiters can also be responsive to the “Document Language” setting in a bilingual document.

If the “Document Language” setting is English, the [Begin_English] … [End_English] section is retained and the [Begin_Chinese] … [End_Chinese] section is repealed; and vice versa if the “Document Language” setting is Chinese.

Example：

[Begin_English]
The Company was incorporated on @cIncorpDate().
[End_English]
[Begin_Chinese]
本公司在@cIncorpDate()成立為法團。
[End_Chinese]

Result：

If the “Document Language” setting is English, the Chinese section will be repealed:

The Company was incorporated on 13/09/2018.

If the “Document Language” setting is Chinese, the English section will be repealed:

本公司在 2018 年 9 月 13 日成立為法團。