VB Scripting for CATIA V5: Email Course by Emmett Ross
Lesson #1 - Introduction to CATIA Programming

Are you tired of repeating those same time-consuming CATIA processes over and over? Worn out by thousands of mouse clicks? Don't you wish there were a better way to do things? What if you could rid yourself those hundreds of headaches by teaching yourself how to program macros while impressing your bosses and co-workers in the process? **VB Scripting for CATIA V5** is the ultimate guide to teach you how to write macros for CATIA V5!

Through a series of example codes and tutorials you'll learn how to unleash the full power and potential of CATIA V5. No programming experience is required! There are many CAD engineers, designers, and technicians who want to write macros but simply don't have the time or money to go to an expensive third party training class. This text will cover the core items to help teach beginners important concepts needed to create custom CATIA macros. More importantly, you'll learn how to solve problems and what to do when you get stuck. Once you begin to see the patterns you'll be flying along on your own in no time.

What is a Macro and why do we use them?

A macro is a series of functions written in a programming language that is grouped in a single command to perform the requested task automatically. If you perform a task repeatedly you can take advantage of a macro to automate the task. Why do manual labor when you can simply press a button instead? Macros are used to save time and reduce the possibility of human error by automating repetitive processes. Other advantages include standardization, improving efficiency, expanding CATIA's capabilities, and streamlining procedures. Macros use programming but you don't need to be a programmer or have programming knowledge to use them (though it certainly does help).

The application of automation in the design process is virtually unlimited. Some real world examples of CATIA automation at work:

- Batch script for the conversion of CATDrawing files to PDF
- Batch script to convert CATParts to STP files
- Import of points from an Excel spreadsheet to a 3D CAD model
- Export of data from CATIA model to a bill of material spreadsheet
- Automatic geometry creation from selection
- Automatic drawing creation
- Custom functions
- And so on and so on. The possibilities are nearly limitless.

Terms and Definitions

The following is a list of terms and their definitions which will be used frequently throughout this text. It is recommended that you become familiar with them if you aren't already. A quick reference of acronyms is listed in the appendices of this book as well.

Integrated Development Environment (IDE) is a computer application to help programmers develop software and typically consists of a source code editor, debugger, build automation tools, object browser, and a compiler or interpreter. IDEs typically have built-in syntax checkers, color coded schemes, and automatic code completion. The Visual Basic Editor in CATIA and Excel is an example of an IDE.

Graphical User Interface (GUI) is a way for humans to interact with computers with graphical elements such as windows, menus, toolbars, icons, etc. which can be manipulated by a mouse. The VBA editor is a perfect example.

Command Line Interface (CLI) is a way for humans to interact with computers through text only and is accessed solely by a keyboard. The most common example is MS-DOS.

Component Application Architecture (CAA, CAA V5, or CNext) is the **Application Programming Interface (API)** or technological infrastructure designed to support Dassault Systèmes products. It is an open development platform enabling programmers to develop and integrate their own applications for CATIA or other Dassault Systèmes products. CAA V5 is faster and more powerful than VB. It provides access to interfaces not available to Visual Basic but is harder to learn. C++ is the primary language. A single source code is used for both Windows and UNIX. CAA **Rapid Application Development Environment (RADE)** provides a workbench to develop PLM applications using the component object model object oriented programming. CAA is beyond the current scope of this text.

Object Oriented Programming (OOP) is where programmers define not only the data type of a data structure, but also the types of operations, or functions, that can be applied to the data structure. An object in software is a structure that consists of data fields and subroutines or functions. Everything in CATIA is an object; the data fields are called *Properties* and the subroutines and functions are called *Methods*. All the data and functions have owners which are the objects to which they belong. A thorough understanding of OOP is critical to your success in macro programming. More on this later.

Component Object Model (COM) is a Microsoft technology that enables sharing of binary code across different applications and languages. CATIA V5 is COM enabled software. Codes for COM objects or components can be called, initiated, or created at any time because they are stored in DLL files and registered in the Windows registry. If CATIA calls Excel, CATIA is then the client and Excel is the server, or the one that provides *services* to the client.

VB talks to CATIA through **Dynamic Linked Libraries (DLL).** DLLs are compiled files that contain all of the functions that make CATIA V5 perform an action. For example, when you

select the “point” function in CATIA, the program calls a function inside one of the dll files that performs the action of creating a point in the V5 database. These files are both compiled and encrypted (or “mangled”) and are located in the UNLOAD directory for CATIA V5 (*C:\Program Files\Dassault Systemes\B20\intel_1\code\bin*). Encryption is a method by which software companies can ensure that others cannot access the function inside the dlls. You cannot directly call the dlls from outside applications, therefore extra programming needs to be done to allow the dlls to be exposed to Windows and the COM object model. This is done via Type Library Files.

Type Library Files (TLB) are files necessary for exposing functions to Windows by acting as maps which point to the functions inside of the dll files that make CATIA V5 work. The TLB files are also located in the UNLOAD directory for V5. Any external application needs to have access to these files. The complete process is: VB Application -> Type Libraries -> Dynamic Linked Libraries -> CNext. How to create references to type libraries is shown in later chapters of this text. To register type libraries in VBA go to Tools>References and select all the libraries you want to use in the project. For CATIA VB programs all of the CATIA type libraries should be selected. Type libraries are .TLB files

Universal Unique Identifier (UUID) - Every CATPart and CATProduct contains a UUID. Basically, CATIA identifies files based on their file name and their UUID. Where problems occur are when two pieces of data have the same UUID. Compounding the problem, the UUID can’t be viewed or edited with any current CATIA function. There are cases when two files may have different names but share the same UUID. This causes a problem when dealing with **Product Data Management (PDM)** systems, like SmartTeam. It is recommended to create new UUIDs whenever possible. Actions which will **create new UUID** include:

- File + New
- File + New From
- File + SaveAs - option save as new document
- INSERT New Product
- INSERT New Part
- Document Template Creation

Actions which will **keep the same UUID** for each include:

- File + Open
- File + Save Management
- File + Save
- File + SaveAs
- Send to directory
- File + CLOSE
- File + Save
- File + Save ALL

CATIA Macro Languages

Just like most countries on this planet have their own native languages, software programs have their own programming language. Many of these are very similar to each other so learning elements that are common between all programming languages will help you transition from one to another if you need to! For example, after learning to program in CATIA and Excel, I was able (with some help from some tutorials on YouTube) figure out how to program some basic Android applications in Java. That's powerful stuff!

CATIA V5 automation was originally designed for VB6, VBA, and VBScript. Microsoft no longer officially supports VB6 as it has been replaced by VB.net, which is supported by CATIA V5 R16 and onwards. VB6 is more complex but also more powerful than VBA, as is VBA over VBScript and CATScript. Macro languages supported by CATIA and discussed in this text are VBScript, CATScript, and VBA, all derivatives of Visual Basic used in scripting.
CATScript is Dassault Systèmes' portable version of VBScript and is very similar to it. CATScript is a sequential programming language and non-GUI oriented. Regular text editors (like Notepad) can be used for coding. Advantages of writing CATScript macros include free to use, macro recording, personal time saving operations, and rapid deployment. The disadvantages of CATScript are limited flexibility and difficult to debug. The file extension is **.CATScript**.

CATScript macros CAN run on UNIX systems. For CATIA V5 running on UNIX, emulators allow for VBScripts to be run with no interface building tools. Some CATScripts from this text may work under UNIX OS but not all due to differences between the two systems.

VBScript is a subset of the Visual Basic Programming language (VBA). All elements of VBScript are present in VBA, but some VBA elements are not implemented in VBScript. The result of the slimming down process is a very small language that is easy to use. VBScript (officially, "Microsoft Visual Basic Scripting Edition") was originally designed to run in Web applications such as Internet Explorer. One of the advantages of VBScript (in common with other scripting languages) is that it's written in plain, ordinary ASCII text. That means that your 'development environment' can be something as simple as Notepad. CATIA objects can be called but no type is used as the system tries to dynamically call methods and properties of objects. It can be used on both Windows and UNIX versions of CATIA. The disadvantage of VBScript is it's slow, is limited for interface development, and has the least functionality. The file extension is **.catvbs**.

VBScript (MS VBScript) and CATScript are very similar with the major difference being variable declaration. Many programmers believe it is better to declare all variables As String, As Integer, etc. to better keep track of each variable type.

Visual Basic (VB or VB6) is the full and complete version. Derived from BASIC, VB6 programs can generate independent programs, can create ActiveX and servers, and can be compiled. VB programs run in their own memory space.

VBA (Visual Basic for Applications) is another subset of Visual Basic and is hosted in applications such as CATIA (after V5R8), Microsoft Word, Excel, etc. VBA provides a complete programming environment with an editor, debugger, and help object viewer. Declaring the object library used is allowed. In CATIA, VBA has the full VB6 syntax and IDE, which is similar to VBA in Excel. It is event driven, GUI oriented, and has full IDE yet cannot run a program WITHOUT the host application running (meaning it runs as a DLL in the same memory space as the host application). The advantages of using CATvba macros include using the GUI, building forms, and the debugging ability of the macro editor. The disadvantage is VBA programs cannot be compiled into executables or DLLs and they do not run in their own memory space. The extension is **.catvba**.

Visual Basic.NET (VB.net) is Microsoft's designated successor to VB6 and has been supported by Dassault Systèmes since V5R16. VB.net is event driven, has IDE, and is used for building GUI but is not COM (though it can call COM objects). The syntax is different from VB6. Code can be compiled into .exe or .asp files. There are many issues encountered when switching from VB6 to VB.net, such as new syntax, new IDE, new GUI controls, and a new Install Shield, therefore fully automatic conversions are near impossible. Compiled languages like VB.net aren't completely necessary because most automation can be done in VBA. VB.net will not be discussed in this text.

There are two primary methods a macro communicates with CATIA: in-process or out of process.

In Process: The first method a macro communicates with CATIA is when the VB application is ran from within the CATIA process in the computer memory. CATIA essentially freezes while the macro is running and the allocation memory is wiped clean after each run of the program so passing data between subsequent runs is impossible. To access and create in-process macros go to Tools > Macros but please note the only options are VBScript, CATScript, or VBA. Most of the examples in this text are In Process macros.

Out of Process: The other communication method is called out of process where the program runs in its own process in the computer memory. The application could be run from Excel, Word, Windows Explorer, etc. CATIA is fully active while the program is running. VB.net or VB6 can also be used.

How to Create Macros

Macros within CATIA are created by two primary methods:

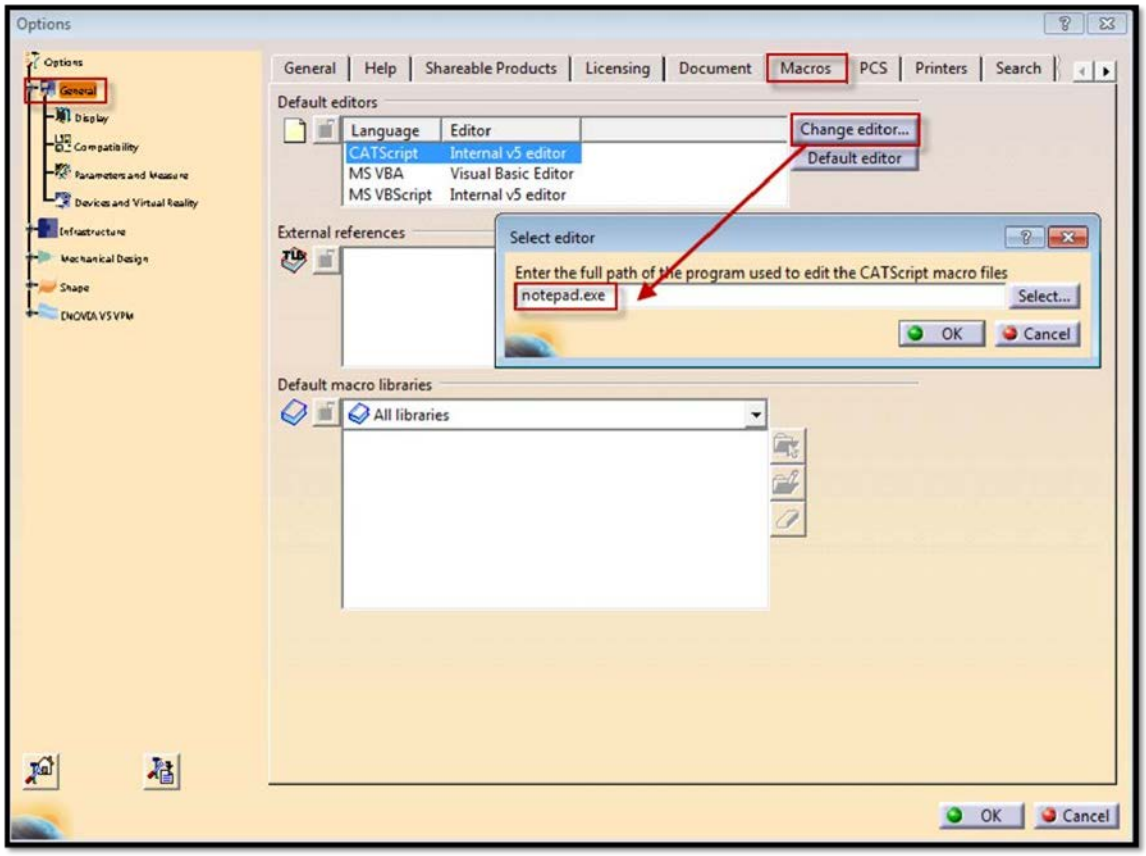
1. Using the macro recorder or
2. Writing custom code with the macro editor

Once a macro is created, there are multiple ways to open the macros window to run your macros:

1. **Go to Tools>Macro>Macros**
2. CATIA macros window keyboard shortcut: **Alt+F8**
3. Assign or create your own icon for each macro
4. Visual Basic Editor (VBE) shortcut: **Alt+F11**

If the macro editor cannot be opened, talk to your system administrator because it may not have been installed. No extra license is required to run macros, though sometimes licenses for special workbenches are needed if the code uses a function or method from a particular workbench.

You can change the default macro editor by going to Tools>Options>Macros>Default Editor. For example, you could change it from the default to Notepad.exe, as shown in the following picture:



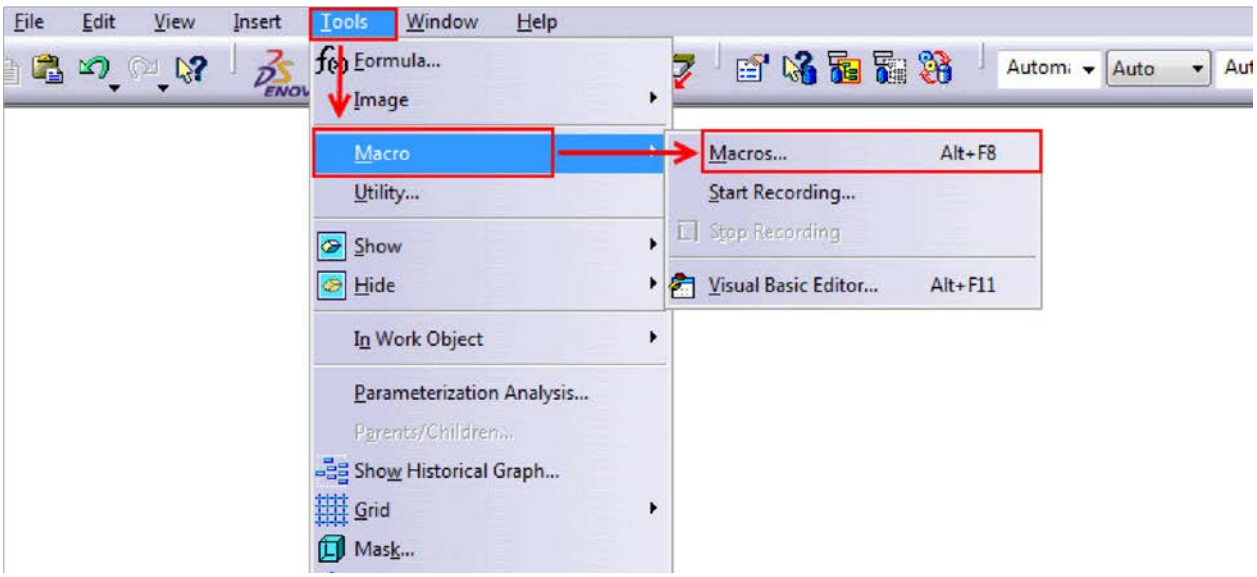
Macro Libraries

CATIA macros are stored in macro libraries in one of three locations: Folders (vbscript and CATScript), Project files (catvba), or documents such as CATParts, CATProducts, and CATDrawings. Only one of these macro libraries can be used at a time. When creating a new macro library, the folder or path location must already exist. If not you will get an error message.

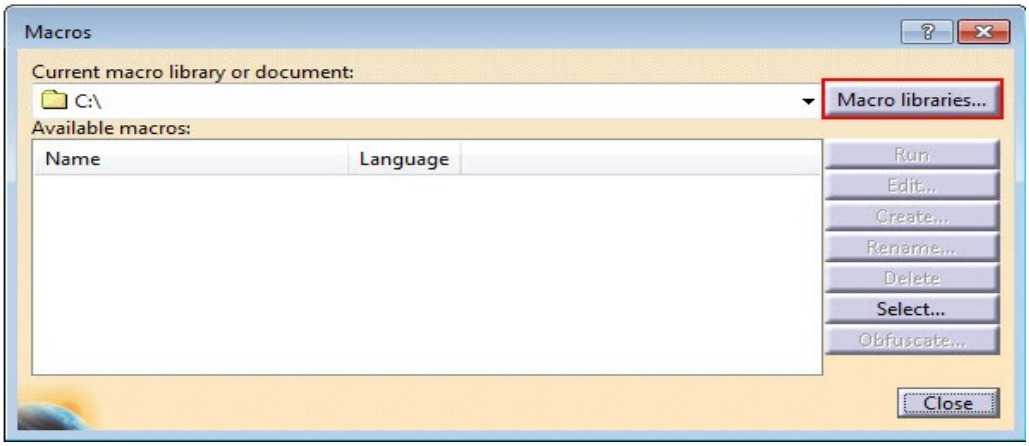
Tutorial 1: Create a new macro library

Use the following steps to create a new macro library or setup an existing one:

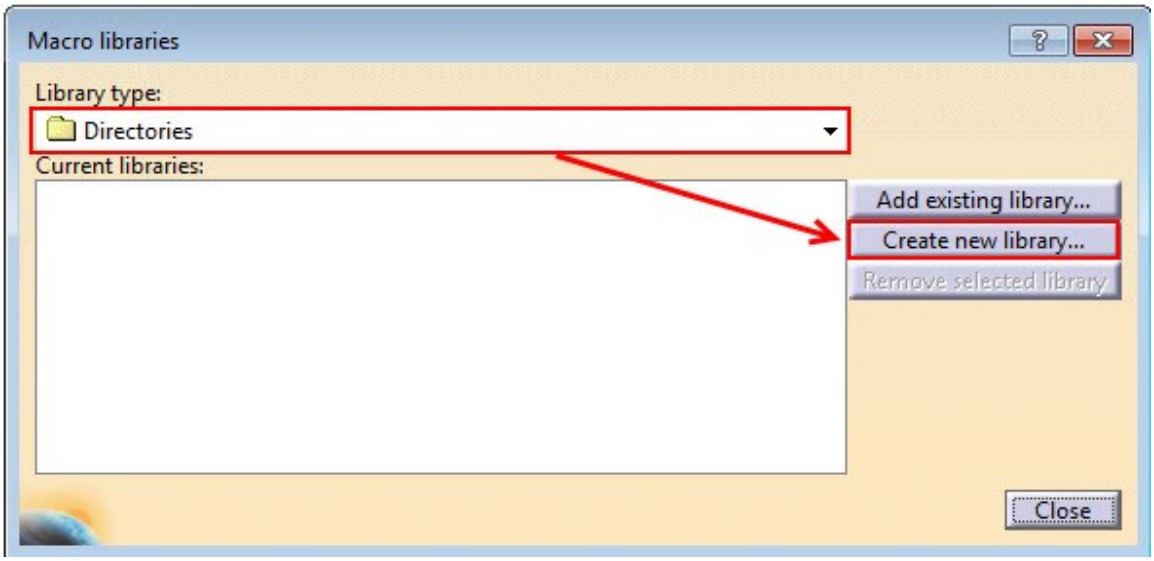
1. Go to Tools>Macro>Macros



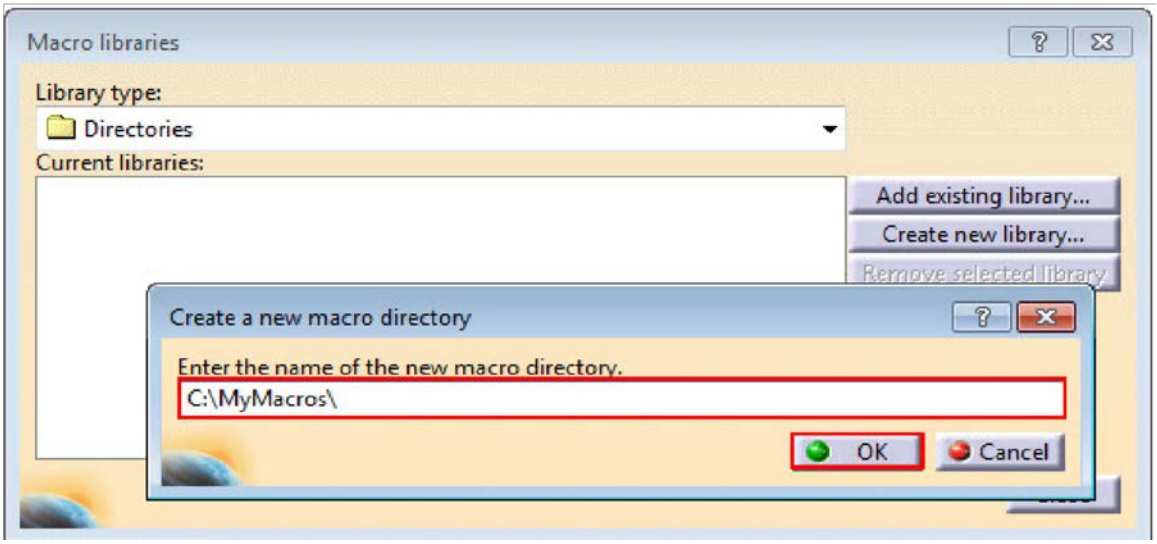
2. Click "Macro libraries…"



3. Ensure the Library type is set to "Directories" then click "Create new library…"



4. Type in the file location where you are planning on saving all of your CATIA macros then click OK.



5. Close the macros libraries window. This is where you can create CATScript macros. If you were setting up an existing library (add existing library versus Create new library) you would see a list of .CATScript files here. You only need to do this once as the library should load even after restarting CATIA.

# # #

Stay tuned for Lesson #2!

Questions or comments? Email me: Emmett@scripting4v5.com

Can't wait for the next lesson? Purchase VB Scripting for CATIA V5 and continue learning today.