

AutoCAD のための VBA の基礎、パート 1 – 2006 年 9 月

「私の将来に対するビジョンは、PC システムが進化して、ユーザがさまざまなアプリケーションを呼び出してドキュメントを生成しているということさえ気付かないレベルまで達するだろうというものです。このビジョンに不可欠な 1 つの要素が、共通のマクロ言語です。共通のマクロ言語によって、いくつかの利点がユーザにもたらされることになります。第 1 にそのマクロ言語は使いやすくなるでしょう。第 2 に、同じマクロ言語をさまざまなアプリケーションで使用できるようになるでしょう。最後に、エージェント-アプリケーションの境界を越えて使用できるグラフィカルインタフェース"機能"-を利用することによって、ユーザはあるアプリケーションで作業しながら、必要に応じて他のアプリケーションの一部を利用したり、あるいは任意のアプリケーションを外部から起動して、複数のアプリケーションをさまざまな方法で結合できるようになるでしょう。」

「したがって、アプリケーションのプログラミングがより一層可能になると同時に、プログラミング言語が全体的に利用しやすくなります。近いうちに、Windows アプリケーションを簡単に、そして楽しみながら作成できるツールが登場することでしょう。」

「アプリケーションのビジュアルコンポーネントをグラフィカルに、つまり、ただ単にコントロールをフォームに配置するだけで設計できるようになることを想像してみてください。すべてのプログラムが、Windows 環境の中で設計、作成、実行されることになるでしょう。BASIC と Windows の両方に共通する使いやすさを取り入れれば、その結果として、現在パーソナル コンピュータに最も普及している環境でアプリケーションを作成するための視覚的かつ生産的で、相互運用が可能なツールが出現するでしょう。」

- マイクロソフト チーフ ソフトウェア設計者ビル・ゲイツ氏

多少の改訂を加えながらも、もともと 4 年にわたって書いてきたこのシリーズ記事が、ビル・ゲイツ氏の引用文と同じように、時間をかけて証明されることを私は願っています。ビル氏は、もはやマイクロソフトのチーフソフトウェア設計者ではありませんが、彼のビジョンは、私たちの仕事を容易にかつ楽しみながら自動化するツールの中にしっかりと生きています。Visual Basic for Applications に照らし合わせて考えてみたときに、ビル氏の引用文がいかに物事を客観的に捉えており、かつ先見の明があったかを、このシリーズ記事を最新リリースの AutoCAD? に搭載されているツールセットに合わせて改訂するにあたって、私は改めて証明することになるでしょう。少し時間を割いてもう一度上記の引用文をお読みになってください。そして、あなたがかねてから疑問に思っていた問題のいくつかの答えがそこにあるかどうか確かめてください。- この記事を読み始める前にそれを確かめてみてください。

二度目に引用文をお読みになった今、あなたはこんな独り言をつぶやいたのではないのでしょうか。「そのとおり！これこそがまさに VBA を学習したかった理由なんだ。よし、さっそく取りかかろう。」さもないければ、あなたはカスタマイズの世界を体験するのは初めてであるため、プロペラ付きの帽子をかぶったり、ポケットプロテクタを身に着けなくても、あなたの仕事をスピードアップするために、ほんの少し何かを覚えたいと思っているだけなのかもしれません。それともひょっとしたら、たまたまこのリンクをクリックしただけなのかもしれません。それでも今では、新しい言語を覚えられるという可能性

に好奇心をそそられているのではないのでしょうか。そうであれば、どの方もここに来て正解です。なぜなら、このシリーズ記事は、そのようなみなさん一人一人のために書かれたものだからです。もし冒頭に示した引用文が 1991 年に書かれたものであると言ったら、あなたはどう思いますか？興味をお持ちいただけましたか？それはよかったです！さあ、始めましょう。

この記事では、以下のトピックについて考察していきます。

- AutoCAD における VBA の歴史
- VBA とは？
- ActiveX とは？
- なぜ VBA を使用した方がよいのか？
- VBA を AutoLISP、Visual LISP、スクリプト、DIESEL などと組み合わせることはできるか？

AutoCAD における VBA の歴史

VBA は AutoCAD のプログラミングリング上では比較的新しい言語ですが、どこかのライト級ボクサーや新人ボクサーのように VBA を一蹴してしまわないでください。VBA (Visual Basic? for Applications)は、BASIC 言語の開発環境を受け継いでおり、その最新バージョンの VBA6 はこれまでの中で間違いなく最も強力です。VBA は 1994 年に Microsoft Excel と Microsoft Project で初めて出現し、AutoCAD には Release 14.0 で"プレビュー版"として導入されました。そして Release 14.01 で、VBA はコアの AutoCAD アプリケーションに永続的に追加されることとなりました。

コンピューティングの歴史において、他のどの言語よりも多くのプログラマが使用してきたこのプログラミング言語が統合されたことによって、カスタマイズと自動化を行うための十分に開発された強力なもう 1 つのインターフェースが AutoCAD に搭載されたのです。Visual Basic for Applications は、そのスタンドアロンの兄弟言語である VB と同じ Visual Basic 言語の構文を利用します。VBA の新しいフォームパッケージ、ActiveX コントロールの完全サポート、そして AutoCAD への完全統合により、VBA は完全かつ強力なものになっています。AutoCAD の内部から使用できる VBA は、インプロセスのコントローラとして機能するため、卓越したパフォーマンスを発揮します。VBA は、ビル・ゲイツ氏の将来のビジョンどおりに、同様の VBA インタフェースを搭載している他のアプリケーションと統合したり、他のアプリケーションを自動的にコントロールすることもできます。

ActiveX とは？

ActiveX オートメーションとは、マイクロソフトが開発したコンポーネントフレームワークの包括的な用語で、アプリケーションにオブジェクトベースのインターフェースを提供します。だじゃれを言うつもりはないのですが、基本的に ActiveX は、VBA を使用して AutoCAD と直接、また比較的理解しやすい方法で対話できるようにするプログラミング(配管)の役割を果たします。これにより、"プログラマ"であるあなたは、プログラミングという複雑なゲーム全体を覚える必要がなくなります。

それがどれほど簡単なものなのかを、次にご説明しましょう。台所に入って、水道の蛇口に手を伸ばすとき、あなたはプラミング(配管)に関する基本的な問題点やルールすべてを理解していないかもしれません。それでもあなたは、そこに流しがあれば、そして水道の蛇口のハンドルを回せば、おそらく蛇口から水が出てきて、床を水浸しにすることなく下水管に排水されるということを知っているでしょう。流しを使用するために配管工である

必要がないのと同じように、VBA を使用するためにプログラマである必要はないのです。オートデスクは、ソフトウェアの配管(ActiveX)を搭載することによって、私たちに大きな恩恵をもたらしました。つまり、自動化の蛇口をひねり、理解しやすい環境で、実に強力なカスタマイゼーションを行うことを可能にしたのです。

オートメーションを使用することにより、AutoCAD の内部で、あるいは Microsoft Excel、Word、Access、Project、Outlook といった VBA 対応の他のアプリケーションから AutoCAD オブジェクトを作成して操作することができます。AutoCAD の ActiveX インタフェースは、クライアントおよびコントローラの両方として機能します。したがって、AutoCAD はコントロールを受けることも、コントロールを行うことも可能です。VBA と ActiveX が一体となって、複数のアプリケーションにわたってプログラミングをコントロールすることができるのです。これは、AutoLISP には存在しない機能です。この自動化のレベルによって、多くのアプリケーションの機能を組み合わせて(配置して)、1 つのアプリケーションで使用することが可能になっています。

なぜ VBA を使用するのか？

初心者、ほんの少しだけキーワードを覚えれば、便利なアプリケーションを作成することができます。さらに、経験豊富なプロフェッショナルであれば、高度なプログラミング言語を使用して実現可能なものはほとんどすべて、この言語のパワーによって実現することができます。かつては"ビジュアルな"アプリケーションを記述するということは、解読するのが困難な膨大な量のコードを記述することを意味していました。これは、アプリケーションを起動したときにその外観をどのようにするべきかを、ただアプリケーションに対して記述するだけのことであっても必要な作業でした。さらに、これには、機能すなわちアプリケーション自体の心臓部であるロジックを実際に行うために記述される膨大な量のコードも加わりました。このようなたてつもない作業が必要になるため、多くの自称開発者は、まさに今から私たちが始めようとしていることを敬遠してきたのです。

VBA の強力なビジュアルインタフェースを使用すれば、ほとんどのプログラミングタスクはバックグラウンドで処理されるため、実際には、構築済みのオブジェクトを画面上の所定の位置にドラッグし、それらを理解しやすいロジックの一部と結合するだけで、きわめて機能性のある便利なプログラムを作成することができます。加えて、AutoCAD ユーザであることが、効果的なユーザ インタフェースの作成に必要なほとんどのスキルを既に持ち合わせていることになります。

VBA の利点として、以下のことが挙げられます。

- 実行時のスピード - AutoLISP アプリケーションに比べて高速です。
- 使いやすさ - AutoCAD に組み込まれており、簡単に使用することができます。
- デバッグ - エラーの検出と修正をこれまでになく簡単に行えます。
- 相互運用性 - Windows に組み込まれているため、VBA を使用して他のアプリケーションとデータをやりとりしたり、他のアプリケーションをコントロールすることができます。
- プロトタイプの迅速な作成 - 最初のフォームを構築してみてください。そうすれば、今までどうして DCL の記述方法の習得に時間がかかったのかと不思議に思うはずです。

VBA を AutoLISP や Visual LISP などと組み合わせることはできるか？

VBA は、AutoCAD 内部で既に使用できる他のプログラミング インタフェースと組み合わせて使用するとベストです。AutoCAD の経験を積んできた方々は、今マウスになじんでいるのと同じくらい、きっとキーボードにもなじめるようになります。Visual LISP には、作成した VBA マクロをロード、実行、アンロードするためのショートカットマクロを簡単に作成できる機能が搭載されています。VBA では、特定のコマンドや関数を実行するために、やはり AutoCAD のコマンド ラインを使用する必要があります、AutoLISP ルーチンと Visual LISP ルーチン両方の呼び出しと実行を完全に行うことができます。AutoCAD の場合はいつものことですが、AutoCAD を自動化したり、そのコントロールをカスタマイズする場合は、現時点において最も簡単に、そして最も直接的に適用できる方法がベストの方法と言えます。このシリーズ記事では、VBA を取り上げて学習しますが、必要に応じて、他のインタフェースも使用していきます。

約すると、オートデスクは、容易に習得できる組み込みの VBA の開発環境を提供しています。また、オートデスクは配管を処理して、VBA を AutoCAD と結び付ける接続も提供しています。VBA は、あなたが使用しているほとんどの Windows アプリケーションで既にサポートされているため、現在は、AutoCAD の内部からそれらのアプリケーションをコントロールすることができます。この機能は双方向性を持っています。なぜなら、現在 AutoCAD は、同様に他のアプリケーションからのコントロールを受けられるからです。本シリーズの次号では、VBA のエディタ（プログラミング ツールボックス）について徹底的に考察しますので、引き続きご注目ください。

本内容は、1991 年『BasicPro』誌に掲載されたゲスト記事から引用したものです。

AUGI 理事会の会長であると同時に、ローカルユーザグループのマネージャでもあり、またフォーラムマネージャでもある Richard Binning による投稿。さらに Richard は AEC/IS Roundtable のメンバーであり、CAD、BIM、および技術関連のブログ *Beside The Cursor* の執筆者でもあります。

Richard は、現在 The Haskell Company に AE テクノロジアプリケーションマネージャとして勤務し、主要なオートデスクトレーニングセンターである Technology Institute of the South のオートデスク認定インストラクタでもあります。