Intelligent Module の作成方法

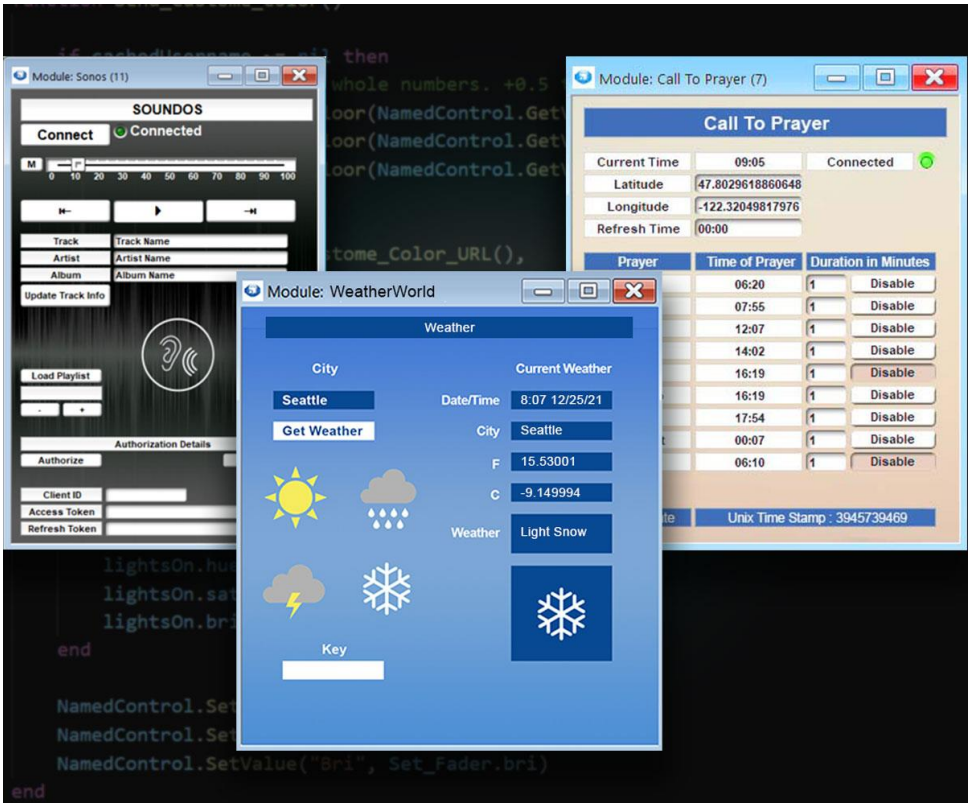
Overview

Intelligent Module は、カスタムコントロールビューと Lua 言語で書かれたスクリプトで構成されています。ユーザーはコントロールビューを操作して、スクリプトの機能を起動します。また、Intelligent Module は、入出力コントロールを通して、ファイル内の他のモジュールと組み合わせて使用することができます。また Lua スクリプトはサードパーティのハードウェアと通信することができ、ユーザーはそのハードウェアの監視と制御を行うことができます。重要なのは、他のモジュールと同様に Intelligent Module のコントロールにリモートコントロールナンバーを割り当てることができ、任意のコントロール画面、つまり T シリーズなどの SymVue デバイスから操作を行うことができます。これは一般的な制御専用機と同様の機能を提供します。Intelligent Module の作成には、若干のスキルや専門知識が必要ですが、Composer のユーザーであれば誰でも利用できるように設計されています

上記のように、インテリジェントモジュールの作成には、Lua スクリプトの知識と、TCP や UDP のネットワークの仕組みを理解することが必要です。本誌では、誰でも Intelligent Module を開発できるように、可能な限り多くの情報を提供するように努めました。Intelligent Module を作成、するために必要なすべての知識が記述されています。

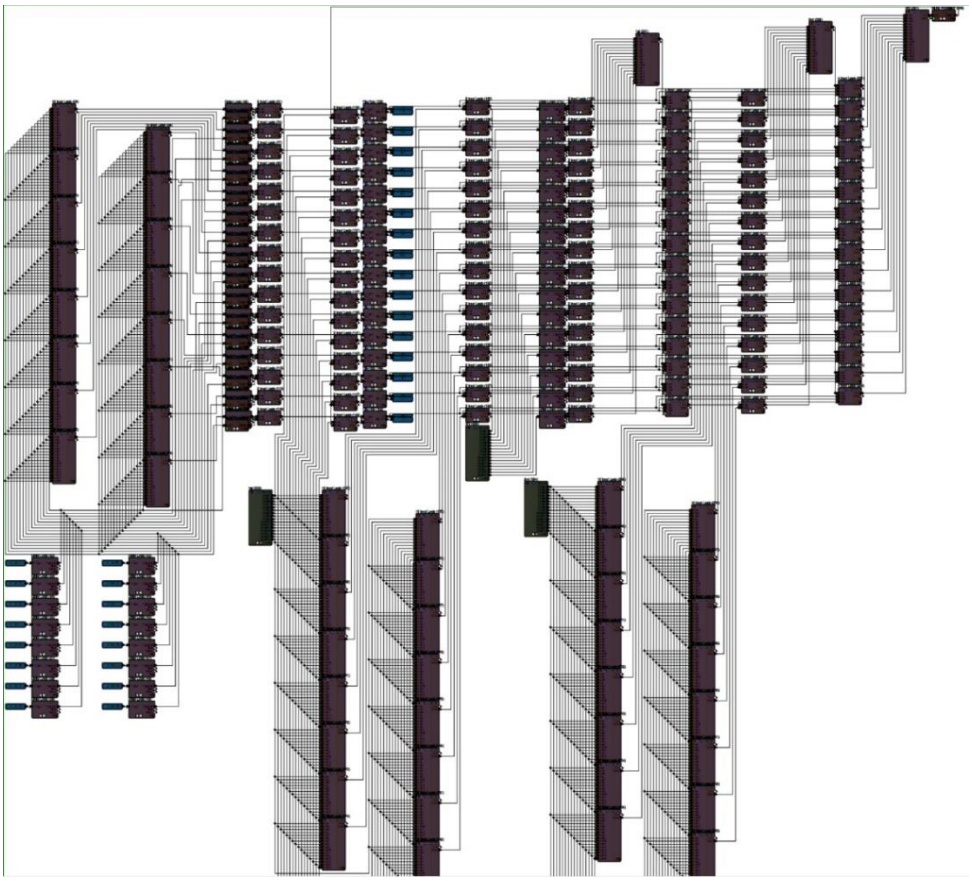
最終的に Intelligent Module を実行するのは DSP デバイスですが、Composer ソフトウェアも Lua スクリプトを実行することができます。そのため DSP が手元にない状態でも Intelligent Module の開発を行うことができます。これは TCP/UDP の制御スクリプトを作成する場合も同様です。Composer ソフトウェアでスクリプトを作成しているコンピュータの NIC を指定し、そこから制御パケットを送信することができます。

Intelligent Module / Lua

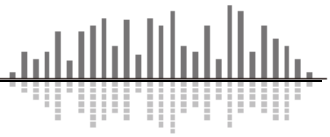


Intelligent Module / Lua スクリプティングと聞くとサードパーティ製品を TCP/UDP で制御することを想像する人が大半だと思いますが、Intelligent Module / Lua スクリプティングで出来ることはそれだけではありません。Composer 内のロジックモジュールで簡単には実現できなかったスイッチングパターンの制御や条件分岐などが DSP リソースを使用せずに簡単に実現することができます。

Composer には、ロジックや制御の仕組みを構築するための膨大な数のコントロールモジュールが用意されています。コントロールモジュールで複雑なロジック回路を作成する場合、多くのモジュール同士を接続する作業が必要となり、面倒な上複雑になればなるほど多くの DSP リソースを消費します。そのため複雑なロジックモジュールを作成した結果、音響モジュールに割くリソースがなくなる、というような本末転倒なことが発生します。

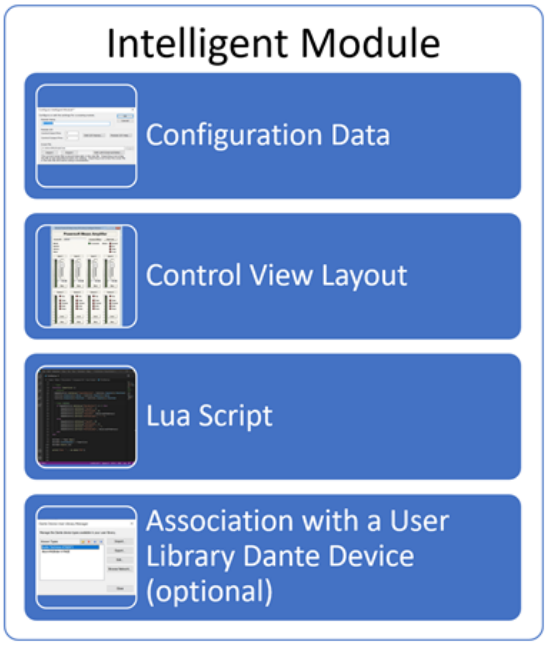


上記の様に何百ものコントロールモジュールを組み合わせてロジックを作成するユーザーも見受けられます。しかし Intelligent Module ではもっとシンプルな数行の Lua スクリプトで同じ結果を得ることができます。また Lua スクリプトは柔軟性が高く変更も容易です。一方コントロールモジュールベースのソリューションは可読性が低く、変更が困難な場合もあります。



INDEX

Intelligent Module の構成



Intelligent Module を作成する時に必要な要素が 4 つあります。

- **設定ファイル（サイトファイル）**

  サイトファイルには、Intelligent Module が含まれます。Intelligent Module にはモジュール名、制御用入出力ピンの番号と名称、Control View Layout、Lua スクリプトファイルとそのパスが含まれます。つまりサイトファイル内には必要な全ての情報が格納されています。これらは、Intelligent Module を最初に作成する際に専用のウィンドウから入力します。Properties パネルから編集することもできます。

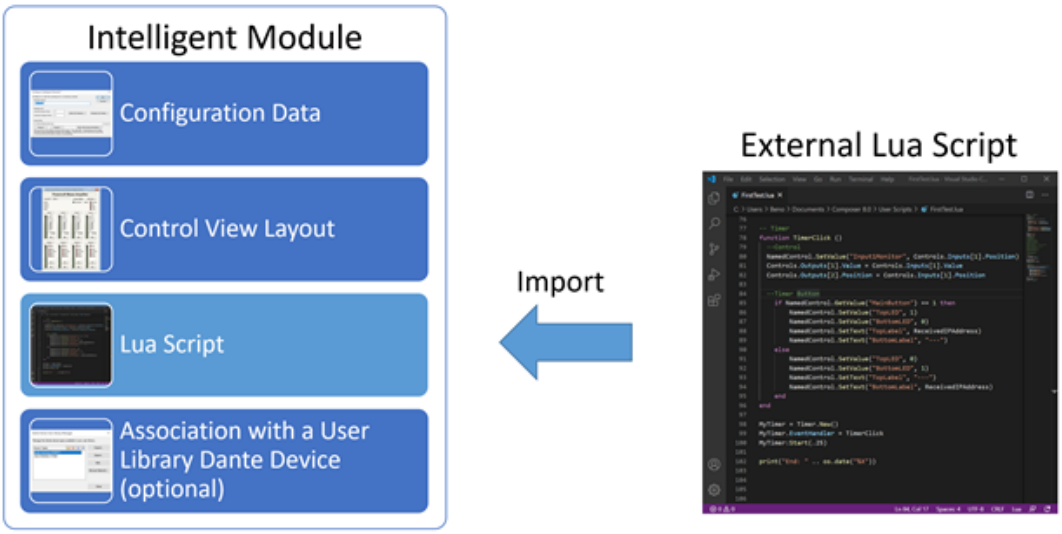
- **Control View Layout**

  Intelligent Module には、フェーダー、ボタン、LED などのコントロールを配置し自由にレイアウトできるコントロールビューがあります。ユーザーは他の Gain や EQ、Compressor などの内蔵モジュールと同様に、これらを操作して Intelligent Module を操作します。これらのコントロールは、制御番号を使用して他のコントロールにリンクし、SymVue で使用するためのコントロール画面に配置することができることが大きな特徴です。

➢ Lua スクリプト

すべての Intelligent Module には、Lua スクリプトが内包されています。このスクリプトは、Intelligent Module の機能を決定します。モジュールのコントロールビュー上のコントロール、入力と出力のコントロールピン、関連するサードパーティデバイスとのデータのやり取りを定義します。

スクリプトは Lua 言語で書かれています。Lua にはコア機能を定義する標準ライブラリがありますが、カスタム機能を追加して拡張できます。これは Lua の強みの 1 つであり、このような用途に非常に人気がある理由でもあります。Symetrix は、Intelligent Module の対話に必要な機能を提供するカスタム Lua API 拡張を作成しました。



上記の様に Composer の外で外部スクリプトエディタを使用しスクリプトを書き、スクリプトが変更され、その結果を確認するたびに、それをサイトにインポートし実行します。このプロセスは非常にシンプルで自動化されており、多くの利点があります。

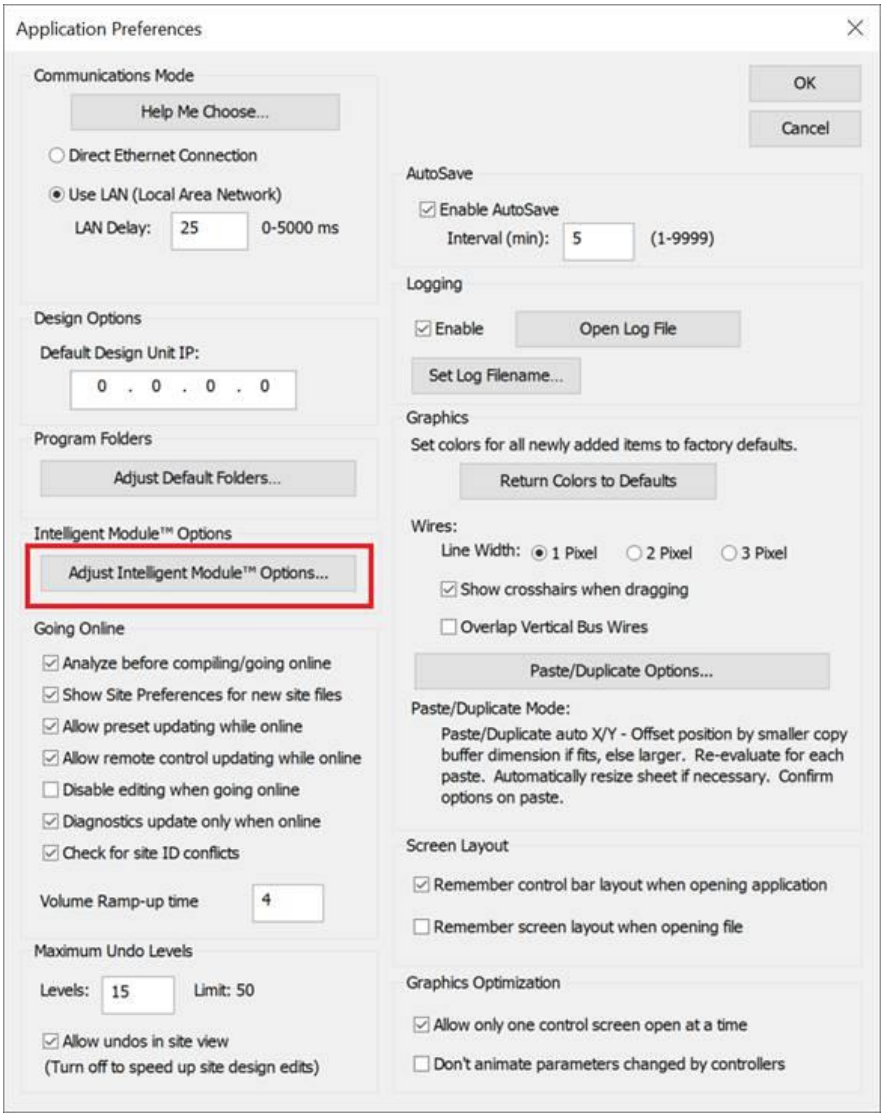
➢ User Library Dante Device との関連付け

Intelligent Modules には上記のスタンドアローン型の他に、ユーザーライブラリの Dante Device に関連付けて使用することができます。これは、Dante Device User Library Manager を使用して行われます。この追加設定により、スクリプトを作成する際に、より優れた機能を利用することができます。

このプロセスについては後ほど詳しく説明しますが、この 2 つのインテリジェントモジュールは、機能および作成ワークフローが異なる 2 種類のインテリジェントモジュールであることを理解しておいてください。

開発環境の準備

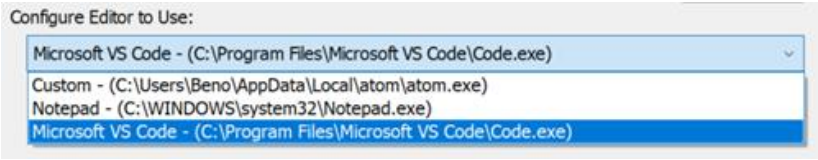
Composer ソフトウェア上の Intelligent Module の設定を行うには Tools メニュー > Application Preferences ウィンドウ > Adjust Intelligent Module Options をクリックします。



この項目では、Lua スクリプトを作成/編集するテキストエディタを選択したり、オフライン動作時に使用する NIC を設定することができます。

テキストエディタの選択

Intelligent Module の Lua スクリプトの作成と編集に使用する外部テキストエディタを設定してください。テキストエディタは Notepad、Visual Studio Code、Custom の 3 つから選択することができます。



- **Notepad（メモ帳）**

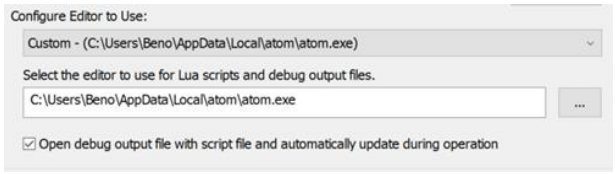
  Windows 標準のメモ帳アプリケーションを使用します。基本的なテキスト編集機能を提供しますが、高度で便利な機能はありません。他のオプションを使用することを強くお勧めしますが、いざというときに利用できます。

- **Visual Studio Code**

  Symetrix は Visual Studio Code の使用を推奨しています。Visual Studio Code は Microsoft によって作成され、無料で提供されています。コンピュータにインストールされている場合は、デフォルトで選択されます。Install ボタンを押すとインストーラーのダウンロードページにジャンプすることができます。本項でも Visual Studio Code を使用し説明します。

- **Custom**

  Sublime Text、Notepad++、Atom などの他の開発用テキストエディタをすでに使用していて、Intelligent Module スクリプトに使用したい場合、このオプションを選択してください。その後、2 つの追加設定が表示されます。

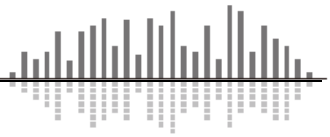


  - **Location**

    最初に使用するエディタの場所を選択する必要があります。“・・・”ボタンをクリックして参照し、選択したテキストエディタの .exe を設定してください。

- **"Open debug output file with script file and automatically update during operation"**

  このオプションをチェックすると、Composer は、開発中に繰り返し行うこの 2 つの作業を自動化することができます。この機能は開発を進める上で非常に重要で、多くの時間を短縮することができます。この機能を使用するには、テキストエディタが開いているファイルが変更された場合に対応できることが必要です。プログラミング用に設計されたほとんどのテキストエディタは、この機能を扱うことができます。しかし、メモ帳のような基本的なテキストエディタでは対応できないため、このオプションを使用することはできません。



外部のスクリプトエディタを使用するメリット

Symetrix がスクリプト作成とデバッグ出力に外部エディタを採用したのは、内蔵エディタやデバッグ出力と比較して明確な利点があり、最も強力な開発環境を提供できるからです。

最も重要なことは、使用するエディタを選択できることです。皆さんの多くは、すでにコードを開発した経験があると思います。私たちは、あなたが最も快適で生産的だと感じる環境で作業できるようにしたいと思います。すでにある特定のテキストエディタに惚れ込んでいる可能性もありますが、Composer ではそれを使うことができます。

また、サードパーティ製のエディタには、強力な機能セットと深いカスタマイズ性があることも利点です。外部エディタでは、スニペット、複数行選択、プラグイン、複数のカラースキームなどを利用して、思い通りの作業を行うことができます。そして、巨大な開発チームに支えられ、定期的にアップデートが行われ、より良いものに仕上がっています。私たちの夢でさえ、マイクロソフトが Visual Studio Code に費やしているようなエンジニアリング時間を割くことはできません。つまり、私たちのエンジニアは、車輪の再発明をする代わりに、製品をより良くするための他の側面に取り組むことができるのです。

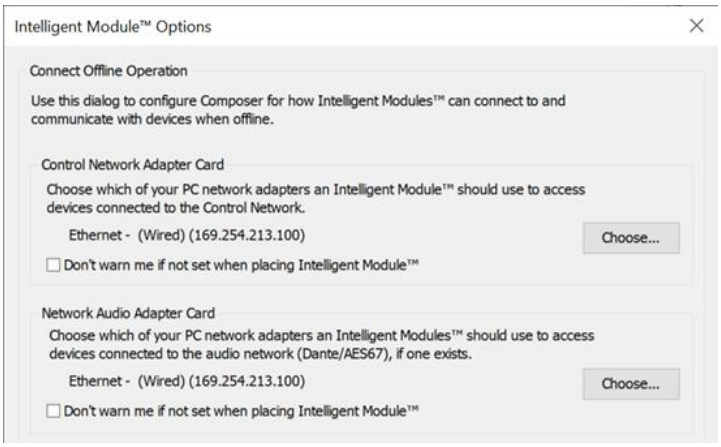
また、スクリプトファイルをサイトファイルの外に置くことで、GitHub のような無料のツールを使って強力なソース管理を利用することができます。これにより、変更をコミットする前にスクリプトが完全に動作していることを確認でき、問題が発生した場合は簡単に以前のバージョンにロールバックすることができます。また、複数のバージョン間の差分を表示し、何が変わったかを確認することもできます。ソース管理は、開発をより簡単かつ安全にするもので、大規模なスクリプトや複数人の開発チームには必須です。

最後に、スクリプト開発を始めたばかりの人は、強力な業界標準のツールを使って学ぶことができるので、スキルを他のプラットフォームに簡単かつ迅速に移行することができるようになります。

DSP がない場合のネットワーク接続

このオプションは、オフラインの Intelligent Module で Composer が使用する NIC を設定します。

つまり DSP がない状態で Composer が Intelligent Module を実行し、その時に使用する NIC を割り当てることができます。



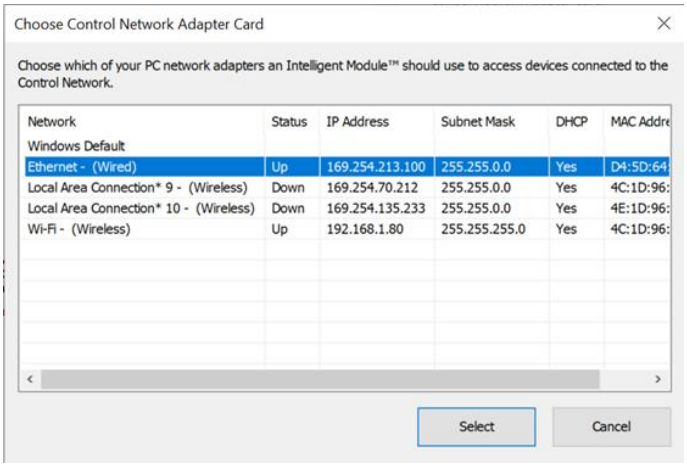
- “Control Network Adapter Card”

  ”Choose”ボタンを押してコントロールネットワークに接続されている NIC を選択します。

- “Network Audio Adapter Card”

  ”Choose”ボタンを押して Dante Network に接続する NIC を選択します。

ポップアップウィンドウが表示され、NIC を選択することができます。



これらが正しく設定されていない場合、Intelligent Modules の様々な部分がオフラインで動作しません。そのため、これらのオプションが設定されていない場合、Composer は Intelligent Modules の配置時に警告を表示します。この警告は、対応するチェックボックスを有効にすることで、ネットワークアダプターごとに非表示にすることができます。

作成のワークフロー

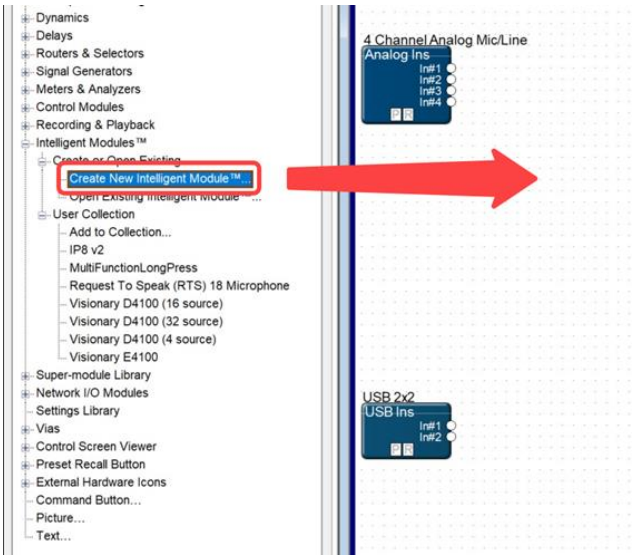
本項では、Intelligent Module を作成するためのワークフローを説明します。Intelligent Module を初めて作成する方は下記の手順に沿って作成することで、作成のワークフローを理解するこができます。

オンライン/オフラインの動作の違いを知るために Symtrix DSP が手元にあった方が理解しやすいですが、なくても作成のワークフローは理解できるはずです。

エディタには Microsoft Visual Studio Code を使用して説明します。

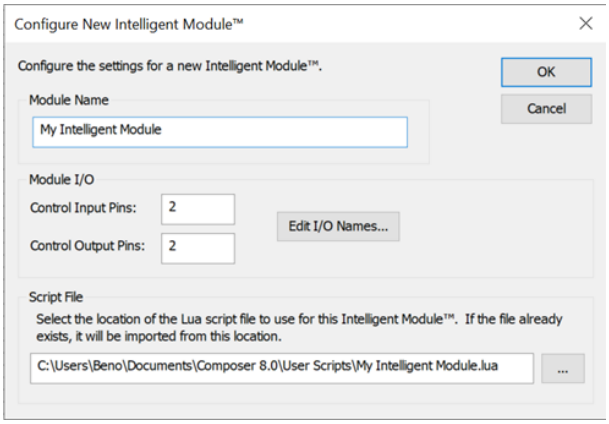
新しいインテリジェントモジュールの作成

新規にオフラインのサイトファイルを作成し任意の DSP を配置します。最初のステップは、新しい Intelligent Module をデザインに追加することです。ツールキットの「Intelligent Module」>「User Modules」で、「New Intelligent Module」をダブルクリックするか、デザインにドラッグします。



New Intelligent Module をデザインに追加すると、Configure Intelligent Module ウィンドウが表示されます。

このウィンドウでは、以下の設定を行うことができます。全ての設定は後から変更することも可能です。



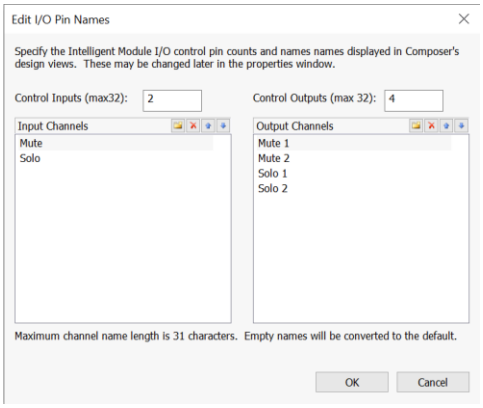
- **Module Name**

  Intelligent Module の名前を入力します。これは、デザインでモジュールを識別するために使用されます。

- **Module I/O**

  この項目はモジュールに必要な制御入力および制御出力ピンの数を入力します。これらは他のモジュールのコントロール入力/出力と同様に使用、接続することができます。入出力が不要な場合は 0 を設定します。それぞれの最大数は 32 です。

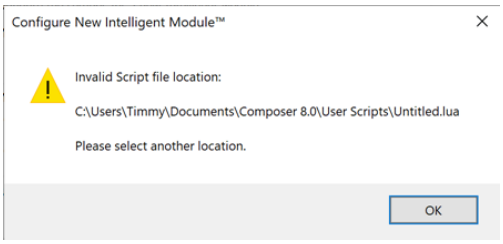
  Intelligent Module の制御ピン数を入力したら、"Edit I/O Names... "ボタンを押し名前を変更することができます。



ここでも、入力や出力の数を変更や、リスト内の名前をダブルクリックして編集することで、それぞれに識別名を付けることができます。矢印は、モジュール上で順序を変更することができます。

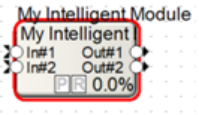
➢ **スクリプトファイル**

すべての Intelligent Module には、Lua スクリプトファイルが内包されています。ここではその関連付けを行います。“・・・“ボタンを使用して、スクリプトファイルの場所を選択します。新規で Intelligent Module を作成する場合は.Lua ファイルを保存する場所を選択してください。既存のスクリプトファイルを選択すると読み込むこともできます。いずれの場合もローカルコンピュータの場所を使用してください。このファイルパスは終了する前に検証されます。パスが使用できない場合は修正するように警告が出ます。



OK をクリックして、Intelligent Module Configuration を完了します。

これで、Intelligent Module を作成する準備が整いました。



これらの設定は、Intelligent Module のプロパティからいつでも再設定することができます。

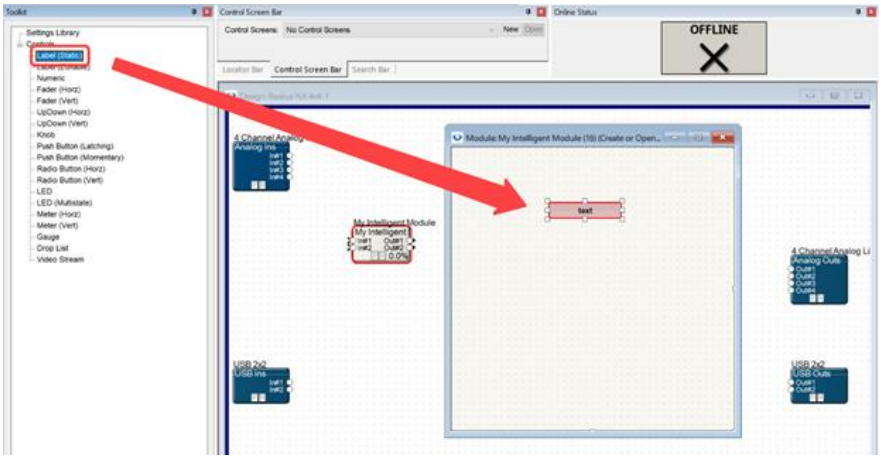
Control View Layout

Intelligent Module をダブルクリックするとコントロールビューが表示されます。

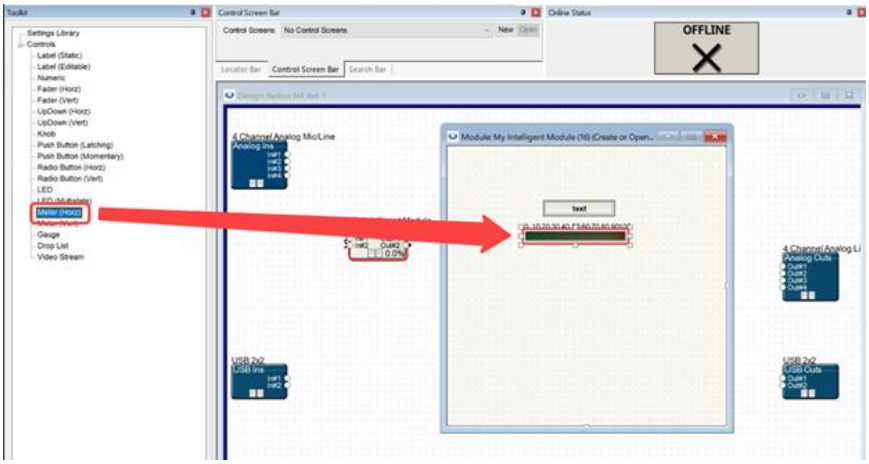
Intelligent Module のコントロールビューを開くと、ツールキットの表示が変ります。

ここでは、まずラベルを追加してみます。

ツールキットのコントロールセクションから「Label (Static)」をドラッグしてください。



次にツールキットのコントロールセクションから「メーター（Horiz)」をドラッグして、メーターを追加してみましょう。



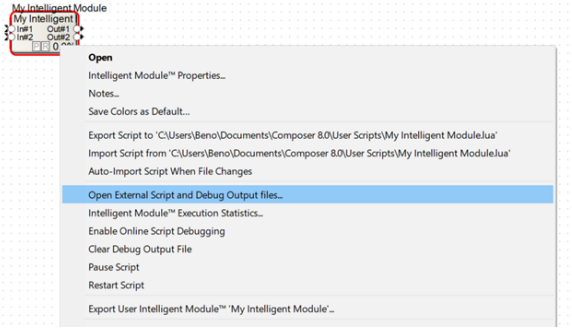
配置したコントロールは、プロパティパネルから設定/変更することができます。

スクリプトの作成

Intelligent Module を作成する時間のほとんどは、Lua スクリプトの作成とそのテスト/デバッグに費やされます。

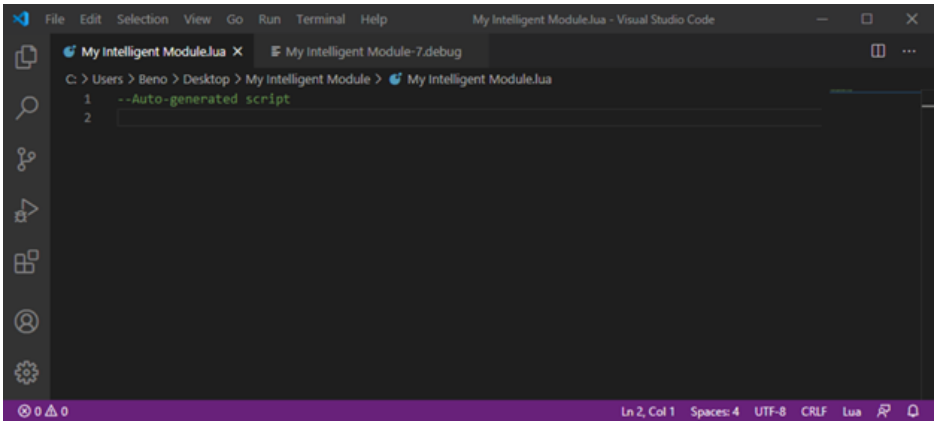
スクリプトを作成するには、Intelligent Module の右クリックメニューのコマンドを多く使用します。右クリックメニューの全体像については後述しますので、ここでは基本的なことを説明します。

まずスクリプトとデバッグ出力ファイルを開く必要があります。右クリックメニューから“Open External Script and Debug Output files..”を選択します。



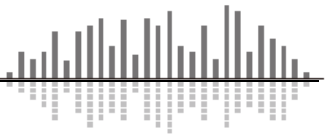
これにより、設定したテキストエディタで、スクリプトとデバッグ出力ファイルの 2 つのファイルが開かれます。

これらのファイルを開くのは今回が初めてなので、Composer は最初の Intelligent Module Configuration で指定した場所に自動的にファイルを作成します。

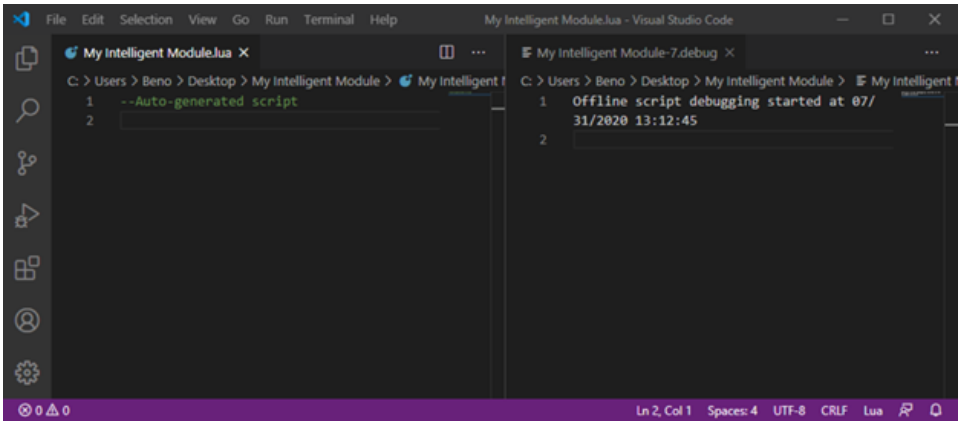


上のスクリーンショットでは、2 つのタブが表示されています。”My Intelligent Module.lua”と名付けた.Lua ファイルとデバッグ出力ファイルです。デバッグ出力ファイルは、スクリプトと同じ名前に Intelligent Module の列挙子を加えたもので、この場合は「My Intelligent Module-7.debug」という名前になっています。

デバッグ出力ファイルは、スクリプトが実行されたときに Composer によって書き込まれます。コンポーザーは自動的にエラーなどを書き出しますが、必要に応じて手動でテキストを書き込むこともできます。一般的には、スクリプトの開発中に、変数の結果や中間状態、スクリプトのどの部分が実行されているかの通知などを出力することが多いでしょう。このファイルは開発者のためのもので、ユーザーがその内容を見ることはありません。



スクリプトとデバッグ出力ファイルを同時に見ることができると便利なので、デバッグ出力ファイルを横にドラッグして 2 列表示にすることをお勧めします（下にドラッグして縦に 2 行表示にすることも可能です）。



スクリプトの冒頭に「--Auto-generate script」というテキストがあることにお気づきでしょうか。これはコメントで、Composer で作成されたスクリプトにはデフォルトで記入されます。またデバッグ出力ファイルには、次のような自動生成テキストが表示されます。

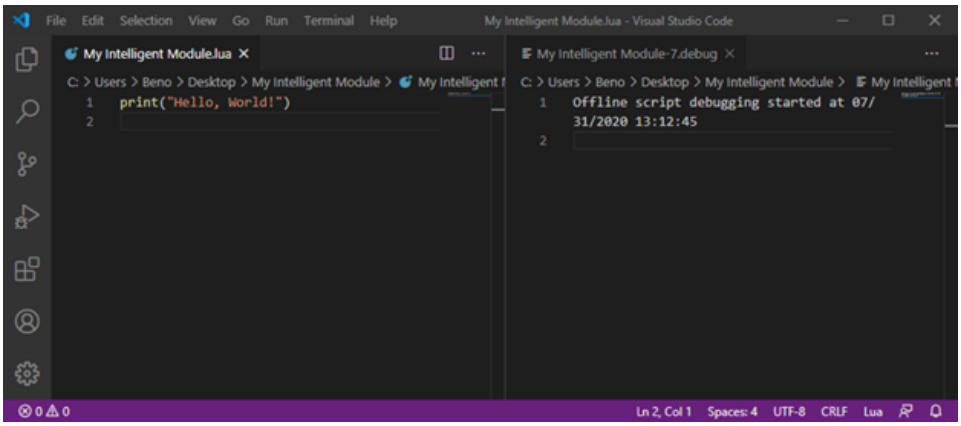
```
Offline script debugging started at 07/31/2020 13:12:45
```

これは、スクリプトがオフラインで実行されているかオンラインで実行されているか？と、スクリプトの実行が開始された日付と時刻を記述しています。この情報は、スクリプトの作成やデバッグ時に必要となることが多いので、自動的に提供されるようになっています。

Hello World!

スクリプトが動作していることを確認し、デバッグ出力ファイルにテキストを生成するために、自動生成されるテキストを何かで置き換えてみましょう。以下の行を追加し、スクリプトを保存してください。

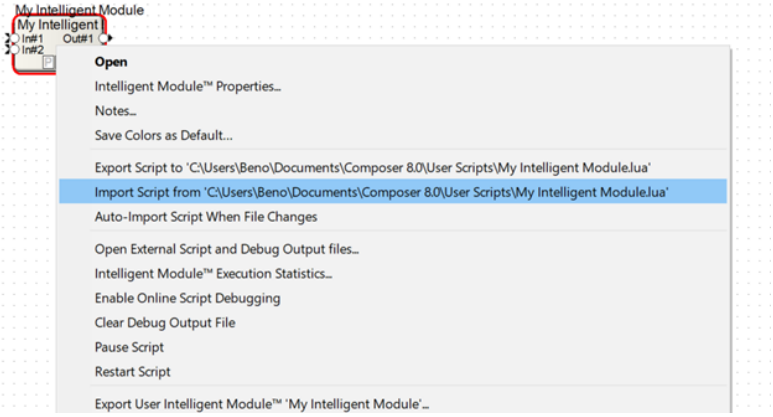
```
print("Hello, World!")
```



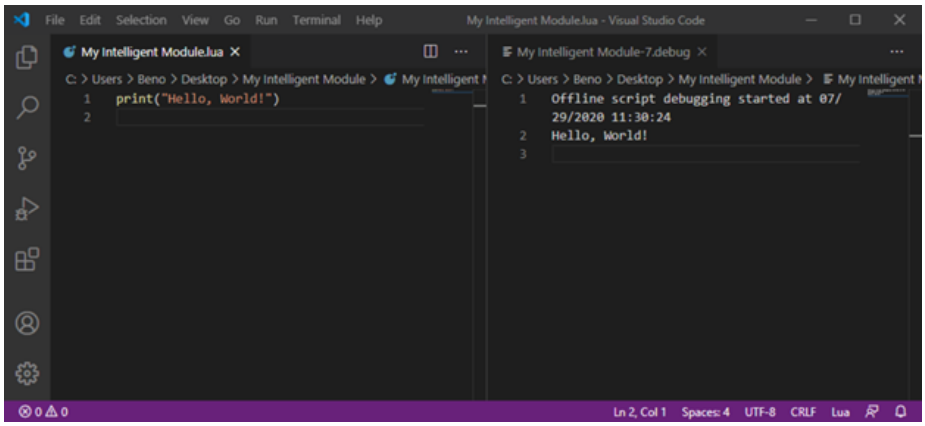
無事入力して保存できましたか？

「Hello, World！」を表示するコマンドを追加しましたが、右側のデバッグ出力ファイルには何も表示されません。これは、スクリプトをサイトファイルにインポートしていないためです。前述のように、Composer は Intelligent Module の実行時に常にスクリプトファイルの内部コピーを使用するため、変更があった場合は必ず外部コピーをインポートする必要があります。

Composer が使用する外部スクリプトをインポートし、その結果をデバッグ出力ファイルで確認するには、Intelligent Module の右クリックメニューから「Import Script from <ファイルパス>」を選びます。



インポートすると Composer がスクリプトを再実行するので、Visual Studio Code を見返すと、デバッグ出力ファイルに「Hello,World!」が Print(表示)されていることがわかります。



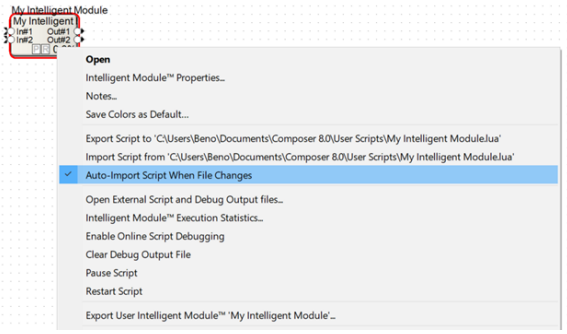
おめでとうございます！

これで、すべてが動作していることが確認できましたので、引き続きスクリプトを作成することができます。

スクリプトを自動インポートする

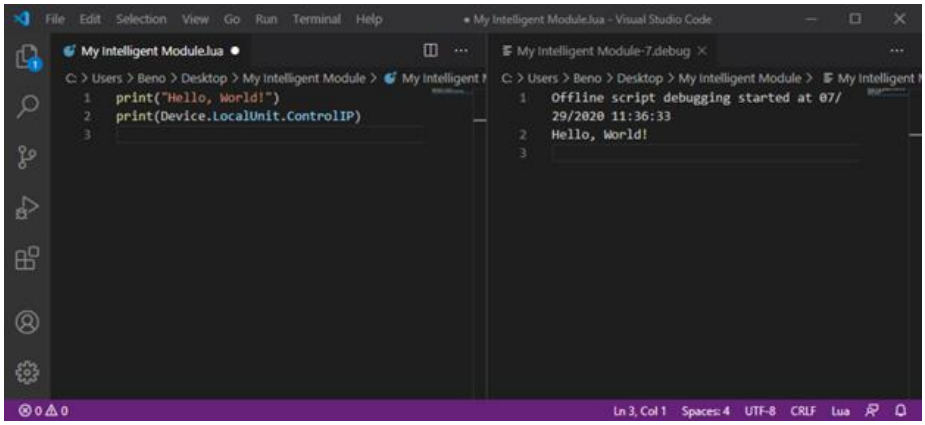
デバッグ出力ファイルの書き込み、インポート、読み込みを繰り返すことになるので、インポートのステップを自動化するオプションが用意されています。

Intelligent Module の右クリックメニューから「Auto-Import Script When File Changes」を有効化します。



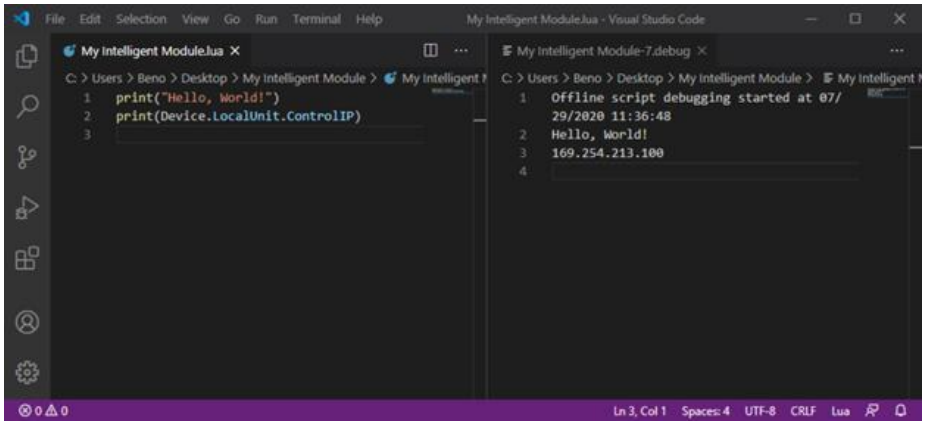
これで、スクリプトファイルをテキストエディタで保存するたびに、Composer は自動的に更新されたスクリプトをインポートし、デバッグ出力ファイルに変更をすぐに確認することができます。スクリプトにもう 1 行追加して、試してみましょう：

```
print(Device.LocalUnit.ControlIP)
```



この行は、現在スクリプトを実行しているデバイスの IP アドレスを表示します。Composer がオフラインの場合は、コンピュータの IP アドレスが表示され、Composer がオンラインの場合は、Symetrix デバイスのコントロール IP アドレスが表示されます。

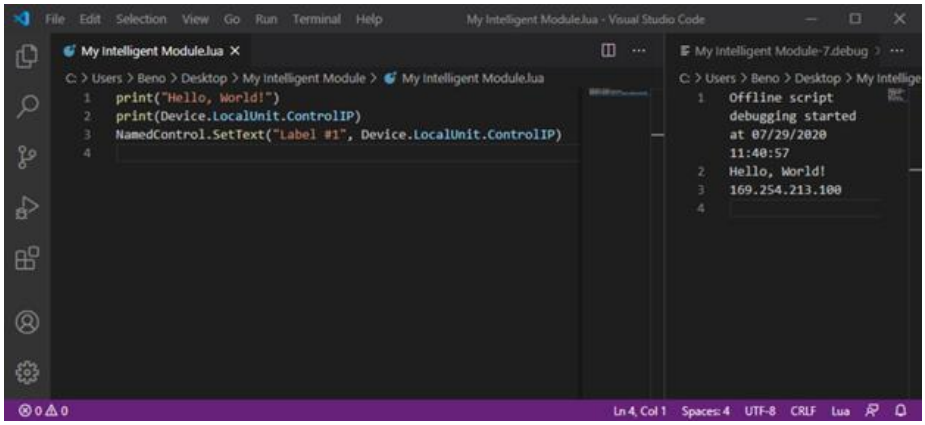
上のスクリーンショットでは、左上の小さな青い「1」  が、スクリプトに変更があることを示しています。保存すれば、Composer に戻って手動で変更を取り込むことなく、すぐに右側の Debug Output File が更新されます。



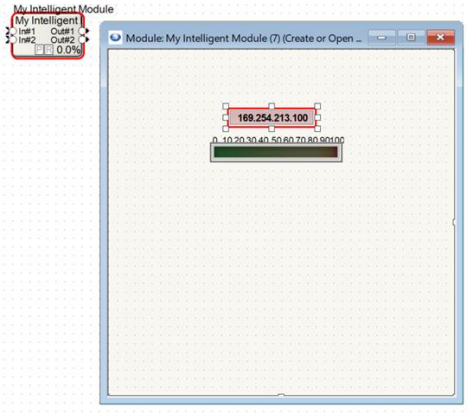
この例では、デバッグ出力ファイルに「169.254.213.100」という追加行があり、私のコンピュータの制御 IP アドレスが表示されていることがわかります。あなたのコンピュータの IP アドレスが表示されているはずです。

デバッグ出力ファイルへの print は便利ですが、この情報をユーザーが見ることができる場所に配置するために、先ほど配置したラベルのテキストを変更しましょう。スクリプトに次のテキストを追加して保存します：

```
NamedControl.SetText("Label #1", Device.LocalUnit.ControlIP)
```



デバッグ出力ファイルは変わりませんが、インテリジェントモジュールのコントロールビューを見ると、ラベルテキストがコンピュータの IP アドレスに設定されていることがわかります。



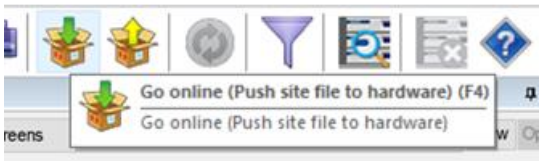
このスクリプトは、「Label #1」というコントロールの Text を Device.LocalUnit.ControlIP の値に設定するよう Intelligent Module に指示しました。繰り返しますが、この時点で上記のコードをすべて理解することを期待するわけではありません。チュートリアルや参考資料で、どのように動作するのかが説明されています。

通常は、オフラインで可能な限り開発を続けてから、オンラインで最終的なテストを行い開発を完了します（オフラインとオンラインのデバッグの違いについては後述します)。このウォークスルーを続けるために、スクリプトを完成させ、オンラインで動作を確認したいとします。

オンラインでスクリプトを実行する

Composer で、「Go online (Push Site file to hardware)」を選択することにしますしオンラインにします。

この作業は本書に興味を持った方なら何度も行った作業でしょう。



さて、オンラインになりました。

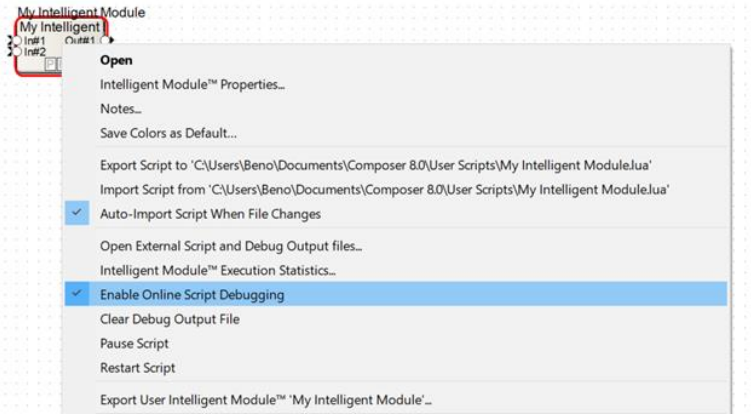
結果は拍子抜けするほど、何も変化がありません。

実際、デバッグ出力ファイルには、まだオフラインであると表示されます。

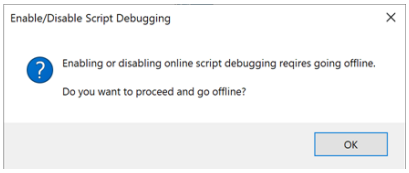
```
Offline script debugging started at 07/31/2020 14:04:12
```

なぜでしょうか？

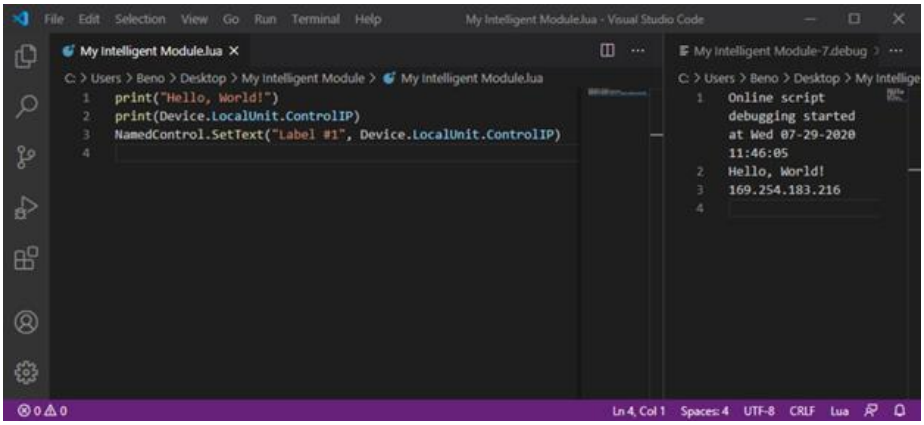
なぜなら、Intelligent Module の右クリックメニューから「Enable Online Script Debugging and Import」にチェックを入れ、Composer と DSP に「このスクリプトはオンラインで実行してください」と明示的に指示する必要があるからです。



この設定を有効にするためには、デバイスに再度ファイルをプッシュする必要があります。そのため、Composer がオンラインのときにこの設定を変更すると、Composer をオフラインにするための警告ウィンドウが表示されます。



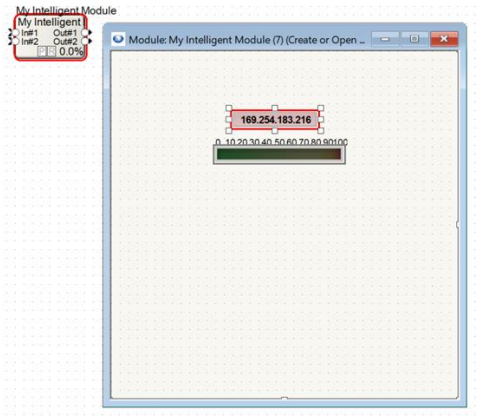
もう一度 Site ファイルをプッシュして Debug Output File の結果を見てください。



デバッグ出力ファイルの上部にある自動情報には、スクリプトが「オンライン」であることが表示されます。

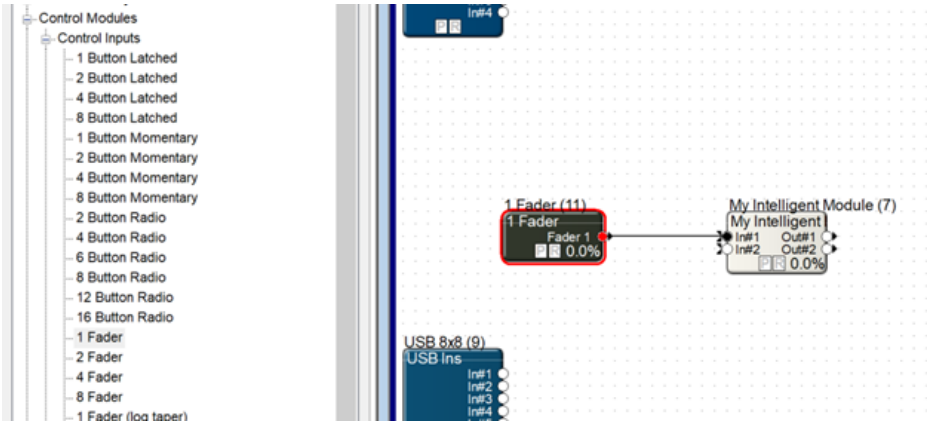
```
Online script debugging started at Wed 07-29-2020 11:46:05
```

IP アドレスはコンピュータのものではなく、ユニットの IP アドレスが表示されていることがわかります。同様に、コントロールビューでは、ユニットの IP アドレスが表示されるようになったことがわかります。



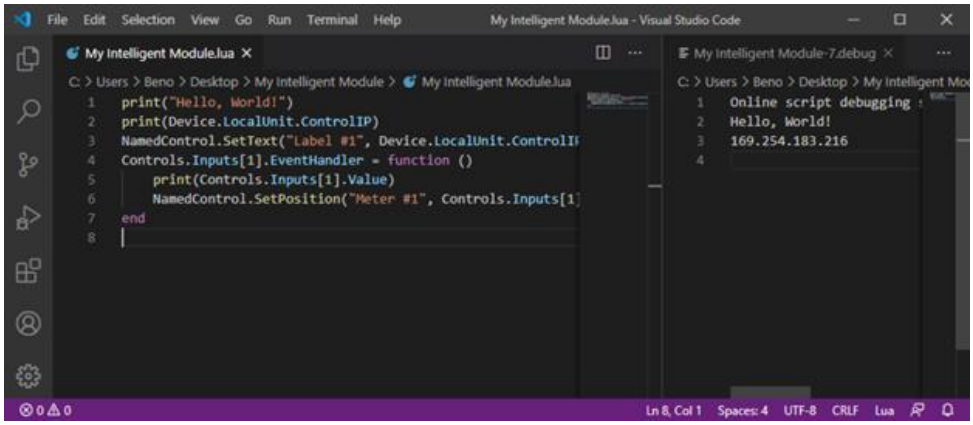
Control Input にフェーダーを接続する

コントロール入力を使ってみましょう。ツールキットの Control Modules>Control Inputs セクションから「1 Fader」モジュールを追加し、Intelligent Module の最初の Control input に接続します。



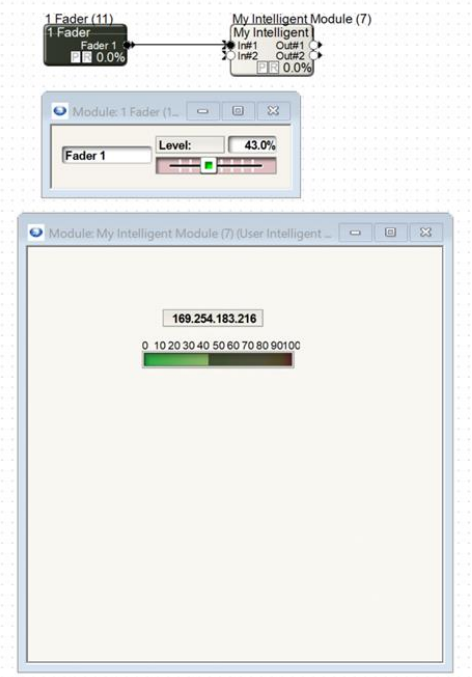
この変更は Composer をオフラインにするので、完了したら、もう一度ファイルをプッシュしてオンラインにしてください。そして、次のコードをスクリプトに追加してください。

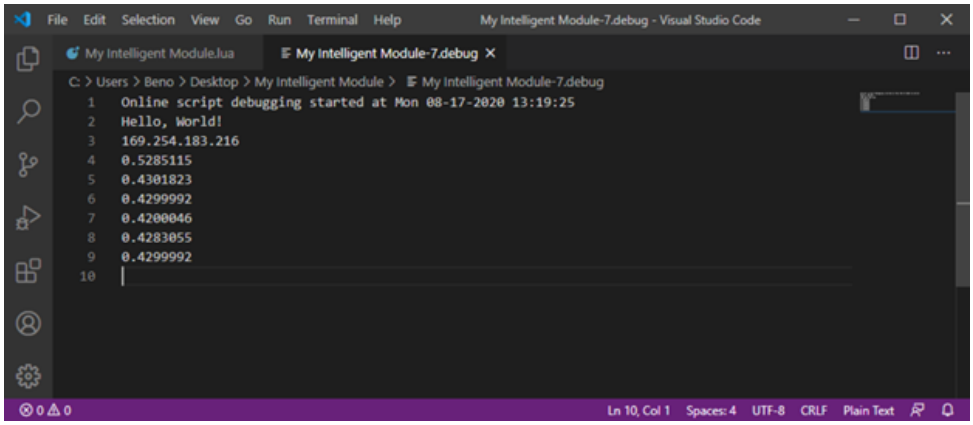
```
Controls.Inputs[1].EventHandler = function ()
    print(Controls.Inputs[1].Value)
    NamedControl.SetPosition("Meter #1", Controls.Inputs[1].Value)
end
```



このコードは、インテリジェントモジュールの最初の制御入力を監視し、それが変化するたびに、関数内のコードを実行します。この場合、1 番目の制御入力の値をデバッグ出力ファイルに出力し、最初に追加したメーターを同じ位置にセットしています。

また、自動インポートとオンラインスクリプトデバッグの両方を有効にしているので、ファイルを保存するだけで、Composer にインポートされ、デバイスに送信されるので、結果を見ることができます。フェーダーを動かすと、Intelligent Module のメーターが更新され、フェーダーの位置が変わるたびにデバッグ出力ファイルに出力されるのが確認できます。

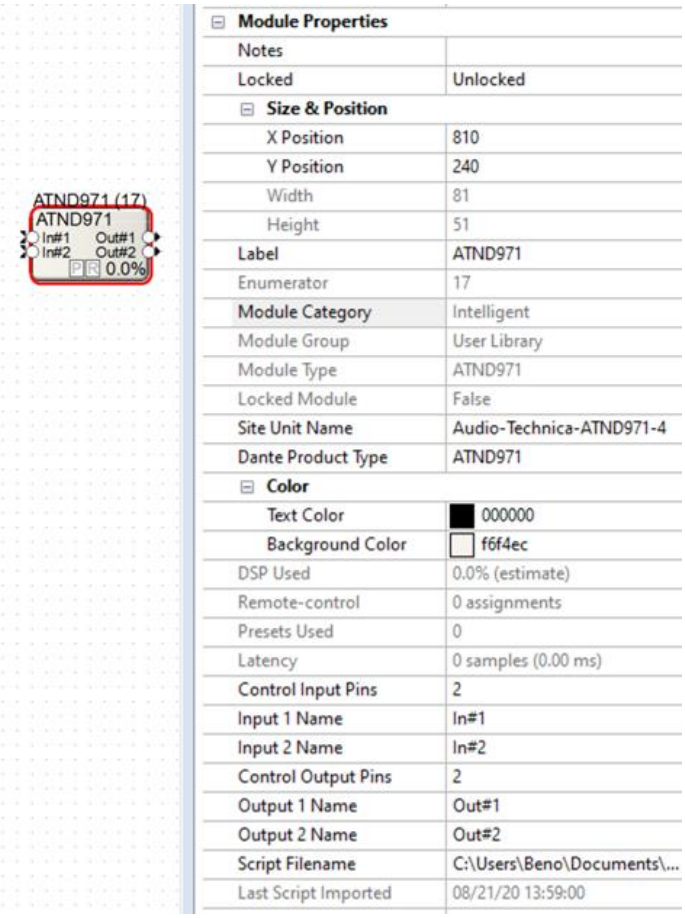




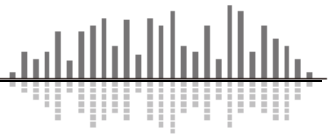
実際の開発では、完成までスクリプトを開発し続け、ハードウェア上での動作を確認してください。

Properties -プロパティウインドウ-

Intelligent Module を選択すると、その情報がプロパティパネルに表示されます。



ほとんどは他のモジュールと同じようなものですが、中にはユニークなものもあります。



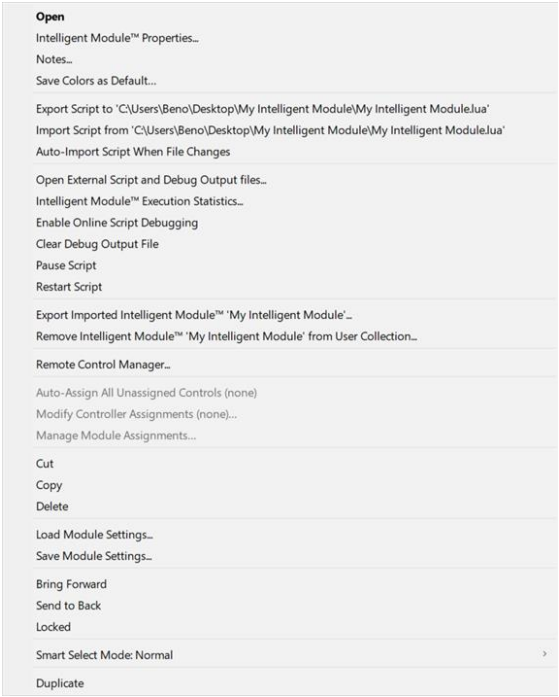
Notes	任意のメモを入力できます。
Locked	デザイン内のモジュール位置がロックされているかアンロックされているかを設定します。 後述の「Locked Module」プロパティとは異なります。
Size & Position	Intelligent Moduleのアイコンの位置を固定します。 Sizeは編集できません。
Label	Intelligent Moduleの名前です。 作成時に名前を入力しますが、ここで変更することができます。 また、スクリプトの初期作成時やエクスポート時にも使用されます。
Enumerator	すべてのモジュールに割り当てられる通し番号で、モジュールを一意に識別することができます。 変更することはできません。これは、スクリプト名とともに、デバッグ出力ファイルに自動的に名前を付けるために使用されます（例：「MyScript-17.debug」）。
Module Category	モジュールのカテゴリーを表示します（この場合は "Intelligent" です）。
Module Group	Module Groupが表示されます。 Intelligent Modulesの場合、以下のGroupが表示されます：
・User Library	ユーザーライブラリーインテリジェントモジュールに表示されます。
・Site	サイト内にのみ存在するスタンドアローンインテリジェントモジュールに表示されます。
・Collection	ツールキットユーザーコレクションに存在するスタンドアローンインテリジェントモジュールに表示されます。
Module Type	Module Typeを表示します。
スタンドアローンインテリジェントモジュールの場合、モジュール名が表示されます。	
ユーザーライブラリーインテリジェントモジュールの場合、Dante製品名が表示されます	
Locked Module	インテリジェントモジュールがロックされているか（true）、ロック解除されているか（false）を表示します。ロックされたインテリジェントモジュールは「L」アイコンを表示し、右クリックメニューが制限されコードを見ることや変更することができません。またロックは解除することはできません。
Site Unit Name	Intellingent Moduleのこのインスタンスに関連する特定のデバイスのサイトビュー名をリストアップします。
Dante Product Type	Intelligent Moduleに関連するUser Library Dante Product Typeを一覧表示します。
Color	
・Text Color	
・Background Color	
DSP Used	Intelligent Moduleは、DSPリソースをほとんど使用しない代わりに、内部のCPUを使用します。 ここにはそのCPUのおおよその使用率が表示されます。
Remote Used	選択したIntelligent Moduleに割り当てられているRemote Control Numberの数を表示します。
Presets Used	選択したIntelligent Moduleに割り当てられているプリセットの数を表示します。
Latency	Intelligent Moduleのレイテンシーを表示します。 Intelligent Moduleはオーディオを処理しないので、常に0を表示します。
Control Input Pins	Intellinget Moduleの入力制御ピンの数を指定します。
Input 1-N Name	Control Input Pinの名称を指定します。
Control Output Pins	Intellinget Moduleの出力制御ピンの数を指定します。
Output 1-N Name	Control Output Pinの名称を指定します。
Script Filename	Intelligent Moduleに関連付けられたLuaスクリプトファイルのファイルパスとファイル名を指定します。別のファイルパスを指定すると、その.Luaファイルをインポートし直すことができます。その場合元ファイルは消えてしまうので注意してください。
Last Script Imported	Luaスクリプトファイルが最後にインポートされた日時を表示します。

右クリックメニュー

Intelligent Module を右クリックすると、様々なオプションが表示されます。これらの多くは、ワークフローの項目で既に説明していますが、ここではその詳細を説明します。これらのオプションは、選択したテキストエディタによって若干異なります。また、これらのオプションは、Intelligent Module がロックされているかアンロックされているかによっても異なります。

Unlock 状態＋Visual Studio Code を使用している場合

Visual Studio Code またはカスタムエディターが選択されていて、“Open debug output file with script file and automatically update during operation”(デバッグ出力ファイルをスクリプトファイルで開き、操作中に自動的に更新する)オプションが**有効**になっている場合、右クリックメニューオプションが以下のように表示されます。



右クリックオプションの機能は以下の通りです。(本書ではわかりやすくするために項目をセクションに分類していますがこれらのタイトルは右クリックメニューには表示されません。)

Standard

Open

コントロールビューを開きます。Module上でダブルクリックしても同様です。

Intelligent Module Propeties

プロパティパネルを開きます。

Notes

他のモジュールと同じように、ノートダイアログを表示します。

Save Colors as Default

すべてのIntelligent Moduleは、デザインビューのカラースキームに同じデフォルトを使用するようにします。

Import Export

Export Sctript to <File Location>

Intelligent Moduleの設定ウィンドウとプロパティパネルで指定された場所に、サイトファイルからLuaスクリプトをエクスポートします。通常新しいIntelligent ModuleのLuaスクリプトを初めて作成するとき、またはサイトファイルまたはIntelligent Moduleファイルを使用してLuaスクリプトを取得するときにのみ実行されます。場所が有効でない場合 (コンピュータに存在しないフォルダなど)、新しい場所を選択するよう促されます。その場所にすでにファイルが存在する場合は、上書きするよう促されます。一度エクスポートすれば、通常、同じコンピュータで再度エクスポートする必要はありません。

Import Script from <File Location>

Intelligent Moduleの設定ウィンドウおよびプロパティパネルで指定した場所から、Luaスクリプトをサイトファイルへインポートします。オフラインの場合は、自動的にスクリプトが実行されます。オンラインの場合で、“Enable Online Script Debugging”がチェックされている場合、スクリプトはデバイスにプッシュされ、インポート時に実行されます。インポート時に、ファイルが存在しない場合、Composerは別のファイルの場所を選択するように促します。

Auto-Import Script When File Changes

スクリプト開発中は、最新の変更結果を確認するためにスクリプトを頻繁にインポートすることになります。このオプションを有効にするとインポートを自動的に実行することができます。

ComposerはLuaスクリプトファイルを監視し、ファイルが保存されるたびに変更されたファイルを自動的にインポートしてスクリプトを再実行します。つまりテキストエディタで保存を押すだけで、変更結果をすぐに確認することができます。**このオプションを有効にすることを強くお勧めします。**

このオプションはIntelligent Moduleごとに有効にする必要があり、Siteファイルを閉じるたびに再有効化する必要があります。

Debugging

Open External Script and Debug Output files…

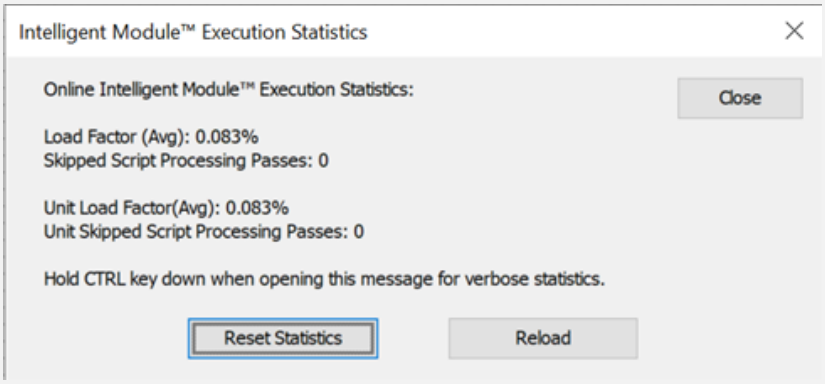
このコマンドは、Luaスクリプトを編集するために最初に実行します。

Composerは、LuaスクリプトとDebugファイルの両方を、設定したテキストエディタで開きます。Luaスクリプトファイルが存在しない場合（新らしいIntelligent Moduleの場合、またはロックされていないIntelligent Moduleを受け取ってそのLuaスクリプトを表示したい場合など）は、Composerは開く前にエクスポートします。デバッグ出力ファイルがまだ存在しない場合も同様に作成されます。

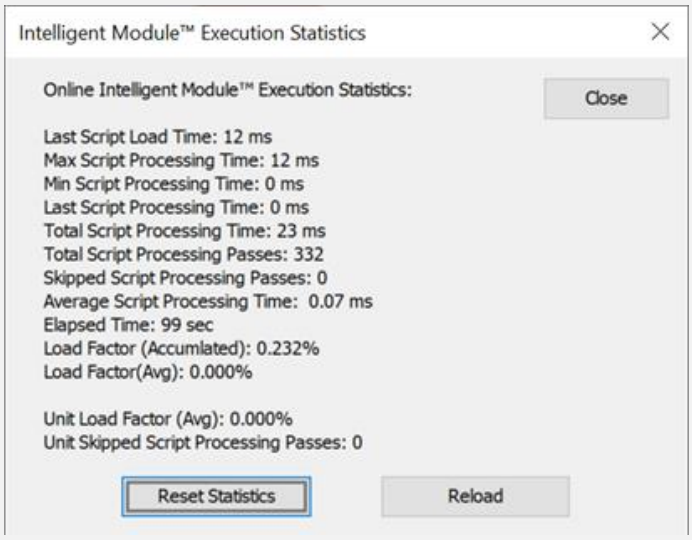
"Auto-Import Script When File Changes" オプションが有効になっていれば、Lauスクリプトの出力をDebugファイルですぐに確認することができます。

Intelligent Module Execution Statistics

Intelligent Moduleの統計ウィンドウを開き、処理時間に関する情報を表示します。



Controlキーを押しながらこの右クリックメニューオプションにアクセスすると、テクニカルサポートから要求される可能性のある追加情報を提供します。

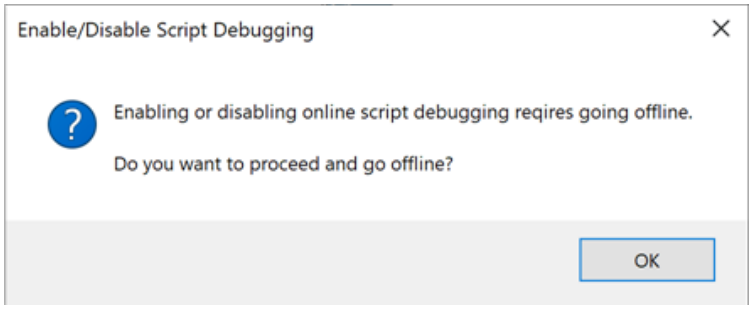


Enable Online Script Debugging and Import

デバッグ出力をオンラインのときに有効にします。

有効にすると、デバッグ出力はオフラインのときと同じようにデバッグ出力ファイルに表示されます。しかし、その裏ではLuaスクリプトがSymetrix Hardwareユニット上で実行されているため、出力は制御ネットワーク経由でComposerソフトウェア(PC)に送られ、それをDebugファイルに書き出します。またこの設定により、インポートしたLuaスクリプトをオンライン機器に自動的にプッシュすることができ、サイトを再度プッシュすることなく、変更したスクリプトの結果を確認することができます。

この設定を有効にするためには、有効にしたあとに再度プッシュする必要があります。そのため、Composerがオンラインのときにこの設定を変更すると、Composerをオフラインにしてサイトファイルとこの設定の状態をプッシュするためのウィンドウが表示されます。

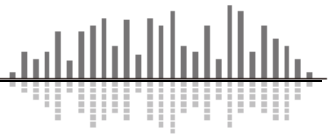


デバッグ出力は、Composerが接続されて出力を見ていない限り、必要のない余分なCPU処理を行います。本環境のIntelligent Moduleをデバイスにプッシュして長期的に使用する前に、このオプションが無効になっていることを確認する必要があります。

これを容易にするために、このオプションはIntelligent Moduleごとに有効にする必要があり、Siteファイルを閉じるたびに再有効化する必要があります。これにより、スクリプトを編集していないときに、不要な、パフォーマンスを低下させるオーバーヘッドが発生しないようになります。

Clear Debug Output File

デバッグ出力ファイルの内容をクリアします。この機能はオフライン時には常に使用可能で、オンライン時には"Enable Online Script Debugging and Import "が有効な場合のみ使用可能です。



Pause Script

スクリプトの実行を一時停止します。これは、スクリプトをデバッグしているときに、デバッグ出力ログの更新を停止させたいときに便利です。このコマンドはオフライン時に常に使用でき、オンライン時には"Enable Online Script Debugging and Import "が有効な場合にのみ使用できます。

この設定はオンラインとオフラインの両方に適用されますが、Composerがオンラインのときにのみ、デバイスの状態を変更します。

たとえば、オンライン状態で、Radius NXでスクリプトが実行されているとします。そこで、スクリプトの一時停止を選択すると、Symetrixデバイスでのスクリプト実行が停止されます。Composerをオフラインにすると、オフラインでの実行はRadius NXでのスクリプト実行と同様に一時停止されたままになります。ここで、スクリプトの一時停止を無効にすると、オフラインでの動作が再開されます。しかし、Radius NX上のスクリプトは一時停止されたままです。最後に、Symetrixデバイスに設定をプッシュすると、Radius NXは一時停止を解除してスクリプトの実行を再開します。

Restart Script

スクリプトの実行を最初からやり直します。このコマンドはオフライン時には常に使用可能で、オンライン時には"Enable Online Script Debugging and Import "が有効な場合にのみ使用可能です。

また、このコマンドは、現在のオンラインまたはオフラインの状態にも影響します。つまり、オフラインの場合、オフラインのスクリプトは再起動されますが、オンラインのスクリプトは実行されつづけます。

Intelligent Module section

Export Intelligent Module <NAME> …

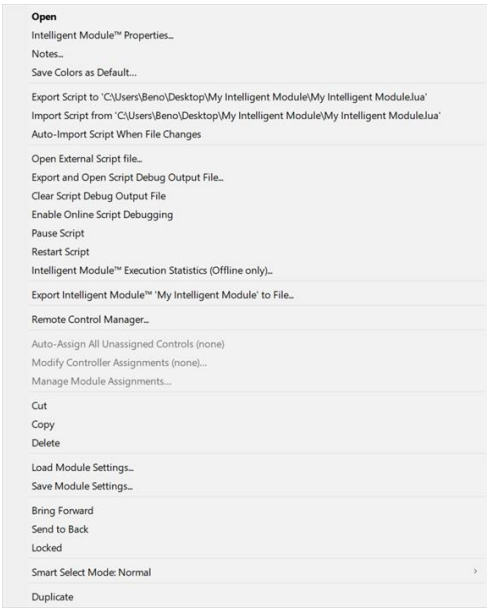
Intelligent Moduleをファイルにエクスポートするための様々なオプションがあるダイアログウィンドウを開きます。Intelligent Moduleがスタンドアローンモジュールか、ユーザーライブラリ・ダンテデバイスに関連しているかによって、正確な機能は異なってきます。

Remove Intelligent Module <NAME> from User Collection…

(ユーザーコレクションのIntelligent Moduleのインスタンスにのみ表示されます) - これを選択すると、インポートしたIntelligent Moduleがツールキットの"Intelligent Module>User Collection"セクションから削除され、今後使用することができなくなります。サイト内で既に使用されている、最近削除されたIntelligent Moduleに基づいたIntelligent Moduleインスタンスは、以前と同じ機能を保持しています。

Unlock 状態＋メモ帳を使用している場合

メモ帳やベーシックなカスタムエディターが選択されていて、“Open debug output file with script file and automatically update during operation”(デバッグ出力ファイルをスクリプトファイルで開き、操作中に自動的に更新する)オプションが**無効**になっている場合、右クリックメニューオプションが以下のように表示されます。



唯一の違いは Debug セクションの最初のオプションにあります。”Open External Script file…”オプションが 1 つではなく、2 つのオプションが表示されます。

Open External Script file…

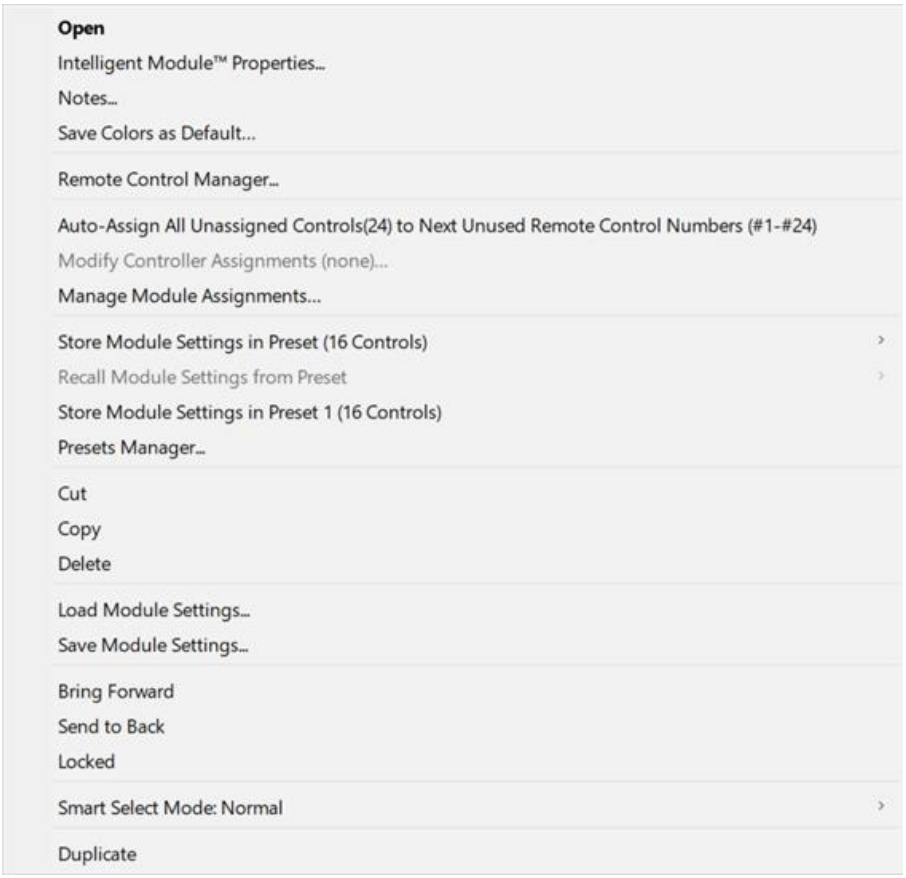
選択したテキストエディタで外部 Lua スクリプトファイルを開きます。

Export and Open Script Debug Output File… -

スクリプト・デバッグ出力ファイルをエクスポートする場所と名前を選択するための” Presents a Save As…”(名前を付けて保存)ウィンドウが表示されます。”Save”をクリックすると、ファイルが保存され、テキストエディタで開かれます。この操作は、デバッグ出力の変更を確認するたびに繰り返す必要があります。一般的なテキストエディタでは、すでに開いているファイルへの変更ができないため、この手順が必要です。Visual Studio Code のようなスマートテキストエディタを使用することをお勧めします。

Lock されている Intelligent Module の場合

ロックされたインテリジェントモジュールをインポートして使用する場合、右クリックメニューは図のように大きく縮小されます。これは、使用するテキストエディタの設定に関係なく、同じ動作となります。



上図のように、スクリプトの管理やデバッグ、エクスポートのオプションは表示されません。

エクスポートとインポート

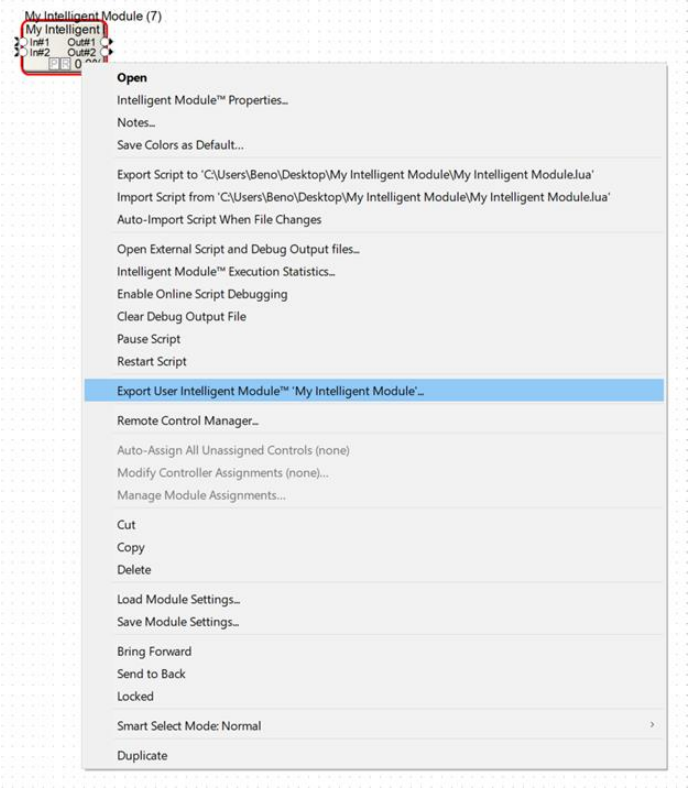
エクスポート

完成した Intelligent Module はファイルとしてエクスポートすることができます。

Lua スクリプトファイルを含む Intelligent Module の全ては.symx 形式のサイトファイル内にも格納されています。しかし.mod や.modx 形式のファイルとしてエクスポートすることで、他のサイトファイルに簡単にインポートすることができるようになります。.modx 形式にすることで、コードを開示できないようにロックをかけることも可能です。

エクスポートした.mod ファイル/.modx ファイルには Lua スクリプトファイルとコントロールスクリーンビューの情報など、Intelligent Module の動作に必要な全てが格納されます。

エクスポートは Intelligent Module の右クリックメニューから「Export Intelligent Module <NAME>...」を選択することで可能です：



選択すると次頁のポップアップが表示され、書き出し方法を選択することができます

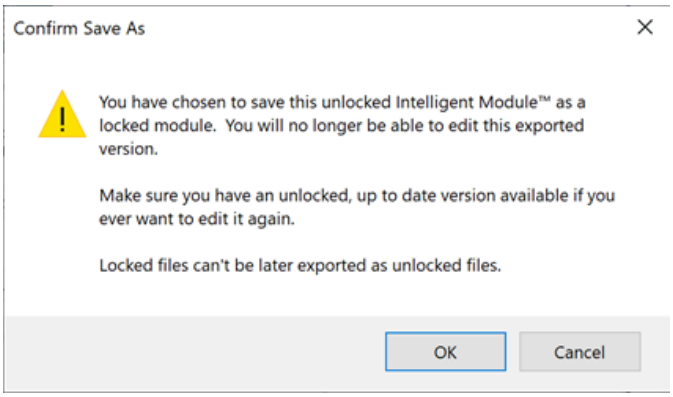
- **Export Unlocked…**

  ロックされていない Intelligent Module を“.mod”ファイル形式でエクスポートします。選択するとファイルの保存場所を選択することができます。.mod ファイルを持つ人は誰でもそれをインポートして、Lua スクリプトを含む Intelligent Module のすべてを編集することができます。これはバックアップを目的とし、将来 Lua スクリプトを編集する必要がある他の人と共有するために使用します。

- **Export Locked…**

  ロックされた Intelligent Module を”.modx”ファイル形式でエクスポートします。選択するとファイルの保存場所を選択することができます。ロックされた Intelligent Module は、通常の使用のみを目的としておりいかなる方法でも再設定/編集できません。コントロールビューのレイアウトの変更や、Lua スクリプトの表示/編集もできません。これは、Intelligent Module 使用する必要があるが、編集する必要がない他の人と共有するために使用するオプションです。

  ロックされた.modx ファイルは、どのような方法でも解除することができないことに注意してください。共有/バックアップにはロックされたバージョンに加えて、ロック解除されたバージョンもエクスポートしてください。エクスポートするときに、このことを説明する警告が表示されます。



インポート

Composer ソフトウェアは.mod/.modx ファイルを簡単にインポートすることができます。インポートには 2 つの方法があります。ファイルを開いて 1 つのサイトで利用できるようにするか、ユーザーコレクションに追加してすべてのサイトで利用できるようにするかです。

Intelligent Module をインポートするには、ツールキットの”from the Intelligent Modules>Create or Open Existing”を選択します。

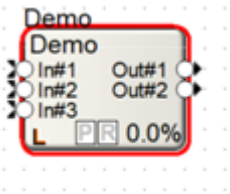
ファイルブラウザが表示されるので、.mod/.modx ファイル選択し開くと、Intelligent Module のインスタンスが追加されます。他のデザインにも同様に追加できますが、Toolkit の Intelligent Modules セクションの User Collection に追加されることはありません。

Intelligent Module をすべてのサイトのツールキットで利用できるようにするには、ツールキットの“Intelligent Modules>User Collection”から“Add to Collection…”を選択します。

同様に、ファイルブラウザが表示され、コレクションに追加する.mod/ .modx ファイルを検索して選択します。これで、Intelligent Module のインスタンスが現在のデザインに追加され、Toolkit に表示されるようになりました。

User Collection に追加すると、全てのファイル上で Toolkit から直接モジュールをドラッグすることができます。ロックされていない.mod ファイルでは、現在のサイトで新規作成した Intelligent Module と同じ機能をすべて備えています。コントロールビューのレイアウトは、スクリプトと同様に編集することができ、同じ Intelligent Module を使用している他のサイトには影響しません。1 つのサイトに同じ Intelligent Module のインスタンスを複数持つことも可能です。

またロックされた Intelligent Module を追加すると右絵のように「L」アイコンが表示されます。



オンラインとオフライン

Intelligent Module は、一般的なモジュールとは異なり、オフラインで使用する際にもある程度の機能が動作します。オフラインでスクリプトを開発・デバッグしているときの動作の多くは、オンライン時と同じですが、いくつか注意すべき違いがあります。

- オフライン開発のメリットとデメリット

  スクリプトは Composer ソフトウェアが動作しているコンピュータで実行され、ネットワーク通信はコンピュータから設定されたポートを介してデバイスに送信されます。ネットワーク構成によっては、Dante と Control のネットワークが別々になっていたり、ブリッジされていたりすることがあります。

  もう一つ重要なことは、オフラインの間は他のロジックモジュールと同様にコントロール信号は動作しません。Intelligent Module の入出力コントロールピンはオンラインの時にしか動作しません。

  オフラインで作業することの重要な利点は、ハードウェアが存在しない状態でスクリプトの大部分を開発できることです。これは、すべてのハードウェアが利用可能になる前に作業を開始できるため、特に重要です。

- オンライン開発のメリットとデメリット

  オンライン中、スクリプトは Sy s sm mmetrix ハードウェア上で実行され、ネットワーク通信はそのハードウェアからコントロールポートを介して他のネットワーク機器に送られます。ネットワーク構成により、Dante と Control のネットワークは別々だったり、ブリッジされている場合があります。

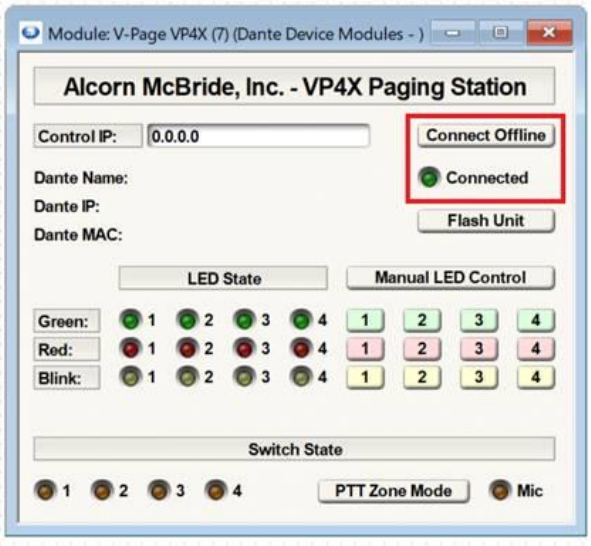
  オンライン状態では、インテリジェントモジュールの全機能を使用でき、コントロールピンを介してデザイン内の他のモジュールと完全に相互作用します

  開発中に新しいインテリジェントモジュールを反復してデバッグするために、デバイスをオフラインにする必要があることです。これは、システムがすでに配備されている場合、時間がかかり、不便になることがあります。

上記の違いを考慮すると、スクリプトのできるだけ多くはオフラインで開発し、最終的なテストをオンライン環境で行うことをお勧めします。

スクリプトは常に実行され続けています

スクリプトファイルは、明示的に一時停止されない限り、オンライン/オフラインに関わらず常に実行されています。オフライン時にスクリプトを停止することを意図している場合は、実行前にオンライン/オフラインの状態を確認するコードをスクリプトに含める必要があります。UDP または TCP を使用してデバイスと通信するスクリプトでは、必要に応じてオフライン操作を無効にするコントロールをユーザーに提供することは良い習慣です。



これは、制御対象機器が 1 度に 1 つしか TCP セッションを開けない場合などに特に重要です。オフラインの ComposerPC と Symetrix DSP 上で動作するスクリプトが同時にデバイスを制御しようとする場合があります。この状況はしばしばデバッグが困難な予期せぬ結果につながることもあります。その場合はオフライン接続を無効にしてみてください。

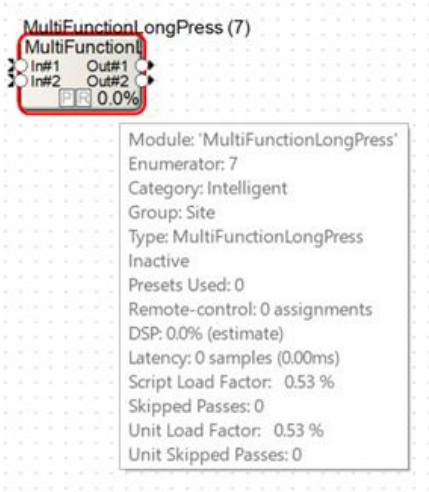
CPU について

Intelligent Module で行われる作業の大部分は、DSP では実行されず、デバイスの CPU で実行されます。デバイスの CPU は、通信や制御、SymVue など、オーディオ処理以外のすべてを管理している CPU と同じ CPU です。当然 CPU の処理能力は限られており、お使いのコンピュータと同じように、同時に実行できるプログラム数は限られています。そのため、Intelligent Module は CPU の処理時間のうち一定の割合しか使用できません。しかしほとんどの場合これを心配する必要はありません。多くの Intelligent Module があり、それぞれが大きなスクリプトを持つ Site ファイルでは、この制限にぶつかることがあります。そこで Composer では、スクリプトが消費する処理量に制限を設け、監視ツールを提供し、制限に近づいたり越えたりした場合に警告を表示します。

Composer とファームウェアは、スクリプトの読み込みやコールバックの実行の各パスの実行時間を測定し、そのすべてが 4Hz（250ms ごと）より速く発生しないようにします。以下の制限が課されます。

- パスの完了に 125ms 以上かかった場合、エラーがデバッグ出力ファイルに出力され、次のパスがスキップされます。
- パスの完了に 250ms 以上かかると、Debug Output File にエラーが出力され、2 つのパスがスキップされます。
- 500ms 以上かかると、スクリプトの実行が阻止され、エラー状態が表示されます。

使用状況をモニターするには、インテリジェントモジュールにマウスを合わせるだけで、次のように表示されます。



- **Script Load Factor**

  ロードファクターは、スクリプトの使用時間をスクリプトをロードしてからの経過時間で割った値で計算されます。その後、過去 40 回のパスを平均し、目標値に対して正規化します

- **Skipped Passes**

  スクリプトの処理時間が長すぎるためにスキップされたパスの数です。

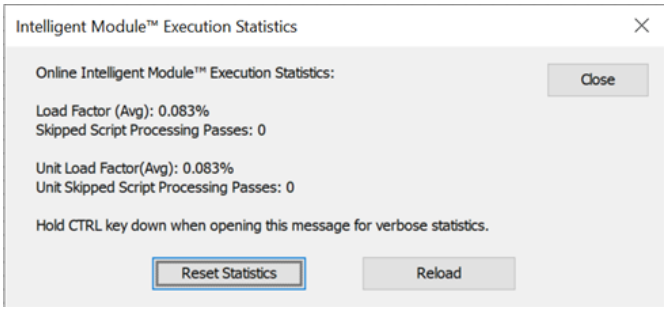
- **Unit Load Factor**

  サイトの同じユニットで実行されているすべてのインテリジェントモジュールのロードファクターの合計値です。これが 100%になると、サイトが CPU 時間を使いすぎていることになります。

- **Unit Skipped Passes**

  サイトの同じユニットで実行されているすべてのインテリジェントモジュールのスクリプトスキップパスの合計です。

この同じ情報は、Intelligent Module の右クリックメニューからアクセスできる”Execution Statistics”からも見ることができます。



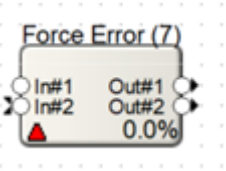
ここでは、統計情報をリセットして、値の計算に使用した測定値を再計算することができます。一般的に、ファームウェアはスパイクを処理することができますが、Unit Load Factor を 100%以下にする必要があります。これにより、プロセッサーが達成しなければならない他の無数の事柄に利用できる十分な処理リソースが残り、パスがスキップされることはありません。Unit Load Factor が 100%を超えていたり、パスがスキップされていたりする場合は、すべての Intelligent Module のスクリプトが処理時間を消費しすぎている可能性があります。個々のスクリプトのロードファクターとスキップパスの数値を使用して、処理時間が最も長いスクリプトを見つけ、その効率を向上させるよう努力します。

これらの数値は、オフラインとオンラインで表示された場合、大きく異なります。これは、ほとんどの場合、PC の方が本体の CPU よりも処理時間が長いからです。CPU 動作の評価をする場合はオンラインで確認してください。同様に、Symetrix の各機器（Radius NX、Edge、Prism、Solus NX）では、インテリジェントモジュールに使用できる CPU 処理能力が異なるため、実行可能なインテリジェントモジュールの数とそれに伴うスクリプトロードファクターは、各機器で異なります。

また、デバッグ出力ファイルへの Print とオンラインスクリプトデバッグの実行は、かなりのリソースを消費します。つまり開発状況ではスクリプトが限界を超えている可能性がありますが、通常の使用では問題なく動作する可能性があることを認識しておいてください。

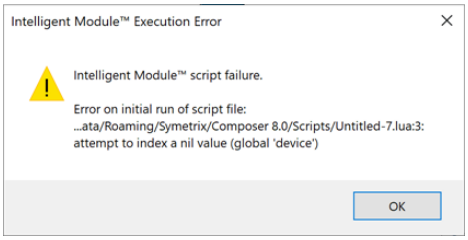
CPU の消費量は同じ実行結果だったとしてもコードの書き方により大きく変化することに注意してください。

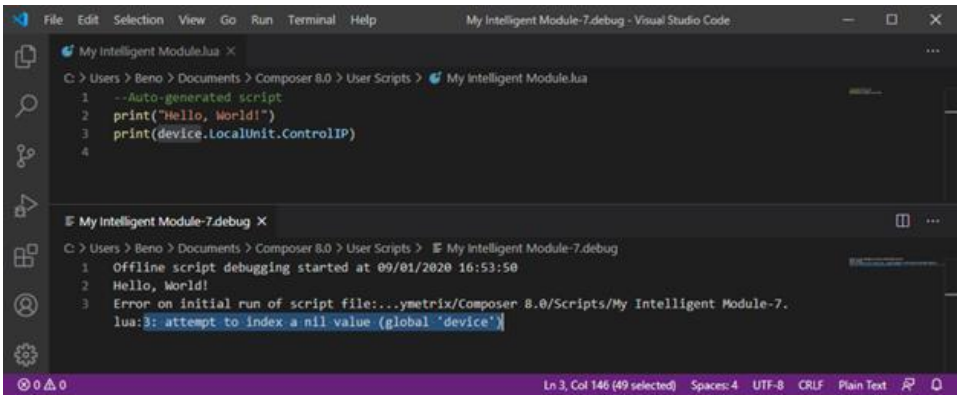
エラー



スクリプトエラーが発生すると、動作は停止しモジュール上に赤い▲マークが表示されます。

Composer ソフトウェアと Symetrix DSP は、Lua スクリプトがインポートされたときに、コードの全てをチェックしているわけではありません。そのため、Lua スクリプトがインポートされるとすぐにエラーになることもありますが、特定の操作が行われるまでエラーが検出されないこともあります。いずれにせよ、コードを間違え Lua インタプリタでエラーが発生した場合、Lua スクリプトの実行を停止してエラーを表示します。モジュールをクリックすると詳しいエラーの情報がポップアップで表示されます。エラー情報はインタプリタから送られてくるもので、同じものがデバッグ出力ファイルにも追加されます。



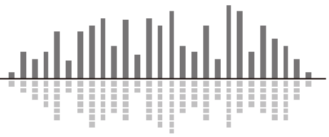


エラーの重要な部分は、行番号とそれに続くメッセージです。行番号はファイル名の後に表示されます（ファイル名とファイルパスが奇妙に見えるかもしれませんが、これは Composer が使用する内部スクリプトファイルなので気にしないでください）。

上の例では、単純なタイプミスで“3: attempt to index a nil value (global 'device') “というエラーが発生しました。これは、エラーが 3 行目にあることを物語っています。そして、メッセージには「device」が nil 値であることが説明されています。これは、大文字の「Device」ではなく、小文字の「d」を使った「device」が使われているためです。Lua は大文字と小文字を区別するので、「device」が何を意味するのかわからないのです。その一文字を変えて再度インポートするだけで、エラーは解消されます。

インタプリタからのメッセージはあまり役に立たないこともありますが、行番号を見れば、何かおかしいところがないか探し始めることができます。エラーのある Intelligent Module があるファイルをプッシュすることはできませんので、まずそれを修正する必要があります。

開発が完了し本環境にて実行中にエラーが出て停止した場合も DSP を再起動する以外に復旧する方法がありません。エラーは開発が完了してからは絶対に発生してはいけません。エラーは通常コードのミスか、実行速度によるものなので、開発中に十分にデバッグする必要があります。



この製品の取り扱いなどに関するお問い合わせは株式会社オーディオブレインズまでご連絡ください。お問合せ受付時間は、土日祝日、弊社休業日を除く 10:00～18:00 です。

株式会社オーディオブレインズ

〒216-0034
神奈川県川崎市宮前区梶ヶ谷 3-1
電話：044-888-6761
https://audiobrains.com

20230426