



Duplex

1. [よくある質問と答え](#)
2. [ステップ・バイ・ステップ・ガイド](#)
3. [アプリケーションの説明](#)
4. [コントロール・マップの説明](#)

1. よくある質問と答え

Q: MIDIコントローラーを使うと、パターン・エディターに沢山のエフェクト・コマンドが勝手に入力されます。

A: それはおそらく、そのMIDIコントローラーがRenoiseのプリファレンス・パネルでもインプット・デバイスとして設定されているからです。メインメニューの [Edit] > [Preferences...] > [MIDI欄] > "In Device A 又は B" で、そのMIDIコントローラーの設定を外してください (Noneにする)。勝手に入力されたエフェクト・コマンドは恐らく、コントロールチェンジ等のMIDI系コマンドでしょう。Duplexは、MIDIコントローラーとアクセスする時には独立したRenoise APIを使うので、Duplexのブラウザ以外でデバイス設定する必要はありません。

Q: Duplexを使って、ノートトリガーする事 (音を鳴らす) は可能ですか？

A: いいえ、出来ません。現状のRenoise APIには "リアルタイム処理" は含まれていません。仮に今Duplexで音を出す事が出来ても、そのタイミングは少々予測から外れます。その為、Renoiseの開発者は現時点ではリアルタイム処理の実装を見送りました。その処理機能が実装されるまでは、コントローラーを (Duplexを使わずに) 直接Renoiseに繋いでノートトリガーしてください。

Q: 私のコントローラーに割り振られたアプリケーションを変更する事は出来ますか？

A: はい。「[デバイス・コンフィグレーションを編集する](#)」事で、アプリケーション・マッピングを変更出来ます。アプリケーションによって、マッピング出来る機能は限定されます。それぞれのデバイス・コンフィグレーションを変更すれば、特定の機能を使う、又は除外する事が出来ます。「[アプリケーションの説明](#)」をじっくり読んで、どんな機能をコントロール出来るのかを知り、アプリケーションのマッピングを考えてください。

Q: アプリケーションのオプションを変更出来ますか？

Q: マトリックス画面でのパターンのトリガーの仕方を選択出来ますか？

Q: ミキサーのPRE/POSTの操作をミキサー・アプリを使って選択出来ますか？

これら全ての質問に対する答えは「[アプリケーション・オプションの変更方法](#)」の解説ページにあります。また、「[アプリケーションの説明](#)」欄を見て、各アプリが持っているオプションを確認してください。

Q: ミキサー/エフェクト・アプリケーションを使って、オートメーションを記録する事は出来ますか？

A: いいえ、現時点では出来ません。現在、これを可能にする方法を模索中です。

Q: 2つのLaunchpadを持っている場合、2つ同時に使用出来ますか？

Q: 同じ種類のデバイスを複数使う事は出来ますか？

A: デバイス・コンフィグレーションを複製し、そのデバイス名の下に2つ目のデバイス名を表示させる事が出来ます。関連する「[デバイス・コンフィグレーション](#)」を開き、必要なコンフィグレーションのコード全体をコピーし、その下にペーストします。そして "display_name" 欄を、例えば "Launchpad (2)" のように変更します。そしてRenoiseを再起動すると、Duplexブラウザのデバイス・リストに複製されたデバイス名が表示されます。新しいデバイスのインプット/アウトプット・ポートの設定を忘れないでください。

例) /Duplex/Controllers/Launchpad/Launchpad.lua を開いて、デバイス・コンフィグレーションの一番下の部分を見てください。そこに "Launchpad (2)" 用のコンフィグレーションが既に複製されてコメント・アウトされています。その部分のコメント・アウトを外すと "Launchpad (2)" がデバイス・リストに加わります。

Q: 同一のデバイスで複数のRenoiseをコントロールする事は出来ますか？

A: 2つのRenoise内で、同一のデバイスを使う事は可能です。しかし、恐らくあなたは、コントロール・マップを2つに分けたいと考えるでしょう。そうしないと、ミキサー・アプリでトラック・ボリュームを動かした時、両方のRenoiseのミキサーが反応してしまいます。デバイスの左半分でRenoise-A、右半分でRenoise-Bという感じでコントロールしたいのではありませんか？ 「[コントロール・マップの分割方法](#)」を読んでみてください。

Q: 全てのデバイスをオフにする方法は？

Q: Renoiseを終了させても、私のコントローラーの幾つかのライトは消えません。

A: Duplexブラウザを開いて、デバイス・リスト内の "none" を選択してください。こうする事で、現在アクティブな全てのデバイスが解除されます。幾つかのコントローラーが適切にシャットダウンし全てのライトが消えるように、デバイス class には特別なコードが含まれています。

Q: RenoiseとMIDIコントローラーの間のコミュニケーション信号を見る事は出来ますか？

A: はい。DuplexとMIDIコントローラーを繋いで（イン/アウト・ポートを設定して）、メインメニューの [Tools] > [Duplex] > "dump MIDI" にチェックを入れてください。実際に送受信されるMIDIメッセージがターミナル（Terminal：端末）に表示されるようになります。（[ターミナルを使用可能にする方法](#)）

Q: "autostart configuration" ボタンをオンにするとどうなりますか？

A: Renoiseを起動するだけで、あらかじめ設定しておいたデバイスが自動的に使えるようになります。

Q: 個別のボタンやスライダーの配置を変更出来ますか？

A: 各アプリケーション毎にアサイン出来るマッピングのセットが設定されていて、その範囲内で変更する事が出来ます。全てのアプリケーションはコントロール・マップのグループを基準としたレイアウトを使っています。ですから、そのアプリケーションの「[マッピング](#)」や「[オプション](#)」を変更すれば、あなたの必要なレイアウトが実現出来るでしょう。
別の方法として、コントロール・マップのグループを移動させる事で、コントロール・マップを再構築する事が出来ます。「[コントロール・マップの分割方法](#)」の説明を読んでください。

Q: 私のデバイスに、さらにアプリケーションを追加するにはどうすればいいですか？

A: もしあなたのコントローラーに未アサインの部分がある場合、又は既に設定されているアプリケーションがあまり重要でない場合、「[デバイス・コンフィグレーションを編集する](#)」事でさらにアプリケーションを追加出来ます。もしデバイス上に追加の余地が無い場合、そのコントロール・マップの為に「[別の分割レイアウト](#)」を作って「[デバイス・コンフィグレーションを編集する](#)」事で、そのアプリケーションを新しいグループにアサイン出来ます。
さらに別の方法として、全く新しいデバイス・コンフィグレーションのプリセットを創り、元のデバイスをそのまま残しておく事も出来るでしょう。

Q: マウスを使わずに私のデバイスのアプリケーションを変更出来ますか？

A: Duplexブラウザを開いてそこにフォーカスしている間は、パソコンのファンクション・キーを使ってコンフィグレーションのプリセットを変更する事が出来ます。その状態でも、他のキー入力はRenoiseのメイン画面に伝わります。

Q: どんなデバイスに対応していますか？

A: 「[Duplexが対応しているコントローラー・リスト](#)」を見れば、現在のデバイス対応状況がわかります。あなたのデバイスが対応済、又は誰かが作業中かどうかわかります。

Q: もっとデバイスを追加するにはどうすればいいですか？

A: もし誰かが作ってくれたデバイス・コントロール・マップと class ファイルをどこかでダウンロードしたなら、そのコンテンツを抜き出し、/Duplex/Controllers/ フォルダ内に追加すれば使えるようになります。詳しくは「[Duplexフォルダの置き場所](#)」の解説を見てください。そのデバイス class の名前と同じサブフォルダを作って、その中にコンテンツを置いてください（もし Monome.lua の場合、フォルダ名も Monome にする必要があります）。

注）もしあなたが自分自身でデバイスを対応させたい場合、次の質問を見てください。

Q: 自分の（未対応の）デバイスのコントロール・マップを作るにはどうすればいいですか？

A: Renoise Forum に新しいトピックを立て、あなたが何を成し遂げたいかを書き込んでください。大抵の場合、コントロール・マップの書き方やそのデバイスの特別な機能の対応方法などを誰かが教えてくれます。また、最初に「[コントロール・マップについて](#)」やデバイス class について学びたい場合、「[Duplex API docs](#)」を見たり、既に対応済のデバイスの class やコントロール・マップを覗いてみてください。

Q: 私はクールなアプリケーションを創りました。どこかで公開する事は出来ますか？

A: まずは、**Renoise Forum**に投稿してユーザーの意見を聞き、そのアプリが関連する全てのデバイス上で動作するか確認してください。そのアプリが十分に役立つ事が確認されれば、公式ディストリビューションに加えられる事になるでしょう。

Q: 私のデバイス・コンフィグレーションをリセット出来ますか？ 標準のセットアップに戻したいです。

A: もちろん可能です。デバイスのインプット/アウトプット・セッティングをリセットするには、**Duplex**フォルダ内の **preferences.xml** を削除してください。もしくは、**Renoise**に公式の**Duplex xrnx** ファイルをドラッグ&ドロップして、デバイス・コンフィグレーションをリセットする事も出来ます。

2. ステップ・バイ・ステップ・ガイド

Duplexフォルダの置き場所

Duplexフォルダは `com.renoise.Duplex.xrnx` という名前で、以下のRenoise Toolsフォルダ内にあります。

- **Windows:** `C:\Documents and Settings\YourName\Application Data\Renoise\V2.6.0\Scripts\Tools\`
 - **OSX:** `Macintosh HD\Users\YourName\Library\Preferences\Renoise\V2.6.0\Scripts\Tools`
 - **Linux:** a hidden folder in the user's home folder: `~YourName/.renoise/V2.6.0/Scripts/Tools/`
-

アプリケーション **class** の置き場所

各アプリケーションは `Duplex/Applications` というフォルダ内にあります。

デバイス・コンフィグレーションの置き場所

デバイス・コンフィグレーション置き場所は、`Duplex/Controllers/[デバイス名]/[デバイス名].lua` です。
（[デバイス名] の部分をあなたのデバイス名に置き換えてください）

コントロール・マップの置き場所

コントロール・マップはXMLファイルで、デバイス・コンフィグレーション・ファイルと同じフォルダに置いてあります。コントロール・マップのファイル名は ".xml" で終わる以外に厳密なルールはありません。しかし、大抵はそのデバイスを象徴する名前、例えば "`Launchpad.xml`" や "`Monome128_StepSequencer`" 等のような名前が付けられています。
（"`Monome128_StepSequencer`" の場合、Monome上でステップ・シーケンサーとして動作するように作られたコントロール・マップです）

もしも、使っているデバイス・コンフィグレーションはわかっているけれど、探しているコントロール・マップが見つからない場合、そのコンフィグレーション・ファイルをテキスト・エディタで開いて "`control_map`" という単語で検索してみてください。

デバイス・コンフィグレーション：アプリケーション・マッピングの変更

各アプリケーション（`Matrix`, `Mixer`...）は、それぞれ独自のアサイン可能なマッピング・セットを持っています。そして実際のマッピングは「[デバイス・コンフィグレーション](#)」内に書き記されています。各マッピングには、その目的に応じた名前が付けられています（"`master_volume for the Mixer`" 等）。ですが、マッピング名は必ずしもRenoiseの機能を表す必要はありません。例えば、もしトロリスのゲームを作ろうと思ったなら、"`rotate brick`"（レンガを回転させる）というようなマッピングを作るでしょう。

まず基本的に、2種類のマッピングがあります。**"indexed"**（索引付きの）マッピングと**"greedy"**（欲張りな）マッピングです。**"indexed"** マッピングは、コントロール・マップ内の個別のインデックスやポジションを指定するのに向いています。それに対して、**"greedy"** マッピングは、提供されたコントロール・マップ・グループ内の全ての利用可能なパラメーターを使い果たすでしょう。**greedy**マッピングの例として、**Matrix**はその基盤の目の様な画面を1つのマッピングでコントロールします。一方**indexed**マッピングは、個別の機能をどこに配置したいかを指定するという意味では、従来のマッピングと似ています。

下の例は、典型的な**indexed**マッピングです。（**Transport** アプリ用の **"play"** ボタン）

```
Transport = {
  mappings = {
    start_playback = {
      group_name = "Controls",
      index = 6,
    },
    [more mappings...]
  }
}
```

マッピング名は **"start_playback"**、コントロール・マップ・グループ名は **"Controls"**、その6番目のパラメーターを指定しています。様々なマッピング名を学ぶには、「[アプリケーション・ファイル](#)」自体を開く必要があります。そして **"self.mappings"** というテキストが最初に出現する所を探します。恐らくそれはファイルの一番最初の方にあるはずです。**Transport**アプリでは以下の様に記されています。

```
self.mappings = {
  stop_playback = {
    description = "Stop playback",
    ui_component = UI_COMPONENT_PUSHBUTTON,
  },
  start_playback = {
    description = "Start playback",
    ui_component = UI_COMPONENT_TOGGLEBUTTON,
  },
  loop_pattern = {
    description = "Toggle pattern looping",
    ui_component = UI_COMPONENT_TOGGLEBUTTON,
  },
  [more mappings...]
}
```

ここで私達が知るべきなのは、そのマッピング名だけです。そのプロパティは内部的に使われるものであって、私達が編集すべきものではありません。

デバイス・コンフィグレーション：アプリケーション・オプションの変更

各アプリケーションは、マッピング・セットを持っているのと同じように、その動作方法を変更するオプション・セットも持っています。それらのオプションは「[デバイス・コンフィグレーション](#)」に書き記され、単純な数値で構成されています。例えば、以下の簡略化したコードは、**Matrix**アプリケーションの **"switch_mode"** というオプションを4に指定して起動します。（マトリックスのパターンの切り替え操作を行った時、次のパターンを予約状態にするオプションです）

```
Matrix = {
  options = {
    switch_mode = 4,
  }
}
```

"4" という番号は突然空から降ってきたものではなく、アプリケーションの特定のオプションを参照しています。どの番号を指定すればいいかは、各アプリケーション・ファイルを開いて知る必要があります。「[アプリケーション・ファイル](#)」を開き、"self.option" と記されたテキストが最初に出現する所を探してください。Matrixアプリでのその部分は以下のようになっています。

```
self.options = {
  switch_mode = {
    label = "When switching",
    description = "What to do when switching from one pattern to another",
    items = {"Stop playback", "Switch to pattern", "Trigger
pattern", "Schedule"},
    default = 2,
  },
  [more options...]
}
```

このように初期設定の数値は "2" です。これは「別のパターンをトリガーすると即座にパターンが切り替わる（switch to pattern）」という動作オプションです。デバイス・コンフィグレーションでこのオプションを "4" に設定すると、「そのアプリケーションの4番目のオプション（上の場合は schedule the pattern）で起動しなさい」という指令を送る事になります。オプション変更後は一旦デバイス・コンフィグレーションを保存してください。そのアプリケーションの次回起動時から新しいオプションが有効になります。

各アプリケーションが持っているオプション・セットの概要は、「[アプリケーションの説明](#)」欄に書いてあります。

コントロール・マップを別々のグループに分割する方法

スペースを使い果たしてしまっていて、他のアプリケーションから機能を追加したい場合、あなたはコントロール・マップを分割しなければならないでしょう。この例として、Launchpad : Matrix + Mixer + Transport のコンフィグレーションを見てください。これはLaunchpadのメイングリッドに2つの分割されたアプリケーションを配置して活用しています。

まずは、「[コントロール・マップ](#)」をテキスト・エディターで開いて、<param> というノードのグループを再編成し、望んだレイアウトになるようにします。この作業には <Column> や <Row> という要素が使えます。もう一度言いますが、Launchpadのコントロール・マップを参照してください。それは、大きなグループを別々の要素に分割するための非常に良い例なのです。

恐らく、望んだレイアウトを作れるまでは少し練習が必要でしょう。ですが、コントロール・マップは読み込みの度に実証されるので、もし何か間違った部分があればその部分を指摘するDuplexからのメッセージが表示されるかもしれません。

Renoiseのスクリプト用端末（Scripting Terminal & Editor）を使えるようにする方法

日本語マニュアルの「[Renoise スクリプトの導入：スクリプト・エディター](#)」を参照してください。

3. アプリケーションの説明

Matrix

MatrixアプリケーションはRenoiseのパターン・マトリックス画面を操作する為のものです。

注) このアプリはグリッドタイプのコントローラー (LaunchpadやMonome等) でしか役に立ちません。

機能：

- up/down/left/right ボタンを使ってマトリックスを操作する
- マトリックス・スロットを押して、ミュート/ミュート解除を操作する
- マトリックス・スロットを押したままにして、そのトラック/パターンにフォーカスを移動させる
- パターン・トリガーを使って、再生順やループ範囲をコントロールする

オプション：

- play_mode - 再生ボタンを押した時の動作を指定する
 - [1:再生/その地点から再生、2:再生/停止、3:パターンの先頭から再生、4:再生予約]
- switch_mode - 新しいパターンに切り替えた時の動作を指定する
 - [1:再生を停止する、2:即座にパターンを切り替え、3:次のパターンを先頭から再生、4:再生予約]
- bounds_mode - "outside bounds" (実際の曲で使われているパターンの範囲外にあるボタン) を押した時の動作を指定する
 - [1:再生を停止する、2:何もしない]
- sequence_mode - パターン・トリガーの動作を指定する
 - [1:複数のボタンを押してループ範囲を操作、2:単純なパターン・トリガー]

Mixer

MixerアプリケーションはRenoiseのミキサー画面を操作するものです。それには全てのトラックが割り当てられ、各トラックのボリューム・パン・ミュート等の現状がページ切り替えによって反映されます。

機能：

- 全てのトラックのボリューム、パンの操作
- 全てのトラックのミュート/ソロの操作
- マスター・ボリュームの独立操作
- PRE/POST ミキサー・モードの切り替え操作
- ページ切り替えによる、各トラックの表示

オプション：

- pre_post - 起動時の PRE/POST ミキサー・モードの設定
 - [1:PRE、2:POST]
- mute_mode - ミュートボタンを押した時、トラックを"MUTE"にするか"OFF"にするかの設定
 - [1:OFFを使う、2:MUTEを使う]
- invert_mute - ミュート状態の表示設定
 - [1:ミュート時にボタンが点灯、2:アクティブ時にボタンが点灯]

- **follow_track** - 現在選択しているトラックのミキサーを表示させたい場合の設定（1列だけのミキサー表示時に必要）
 - [1:Follow track 有効、2:Follow track 無効]
 - **track_offset** - ミキサーのトラック表示をオフセットさせる設定（"2"なら、トラック2からの並びが表示されます。2つのコントローラーを並べて1つのミキサーとして扱いたい場合等に便利）
 - [1:"0"、2:"1"、3:"2"、4:"3"、5:"4"、6:"5"、7:"6"、8:"7"]
-

Effect

Effectアプリケーションは**DSPエフェクター**の全てのパラメーターを操作する事が出来ます。このアプリは現在選択されているエフェクターに追従し、ページ切り替えによってその全てのパラメーターを表示する事が可能です。

機能：

- 全てのエフェクター（プラグインも含む）の全てのパラメーターにアクセス可能
- ページ切り替えによる、各パラメーターの表示
- 固定ボタンによる、各エフェクターの選択

オプション：

- このアプリにオプションはありません
-

StepSequencer

StepSequencerはパターン・エディター上のノート情報を外部コントローラーに表示します。

注）このアプリはグリッドタイプのコントローラー（**Launchpad**や**Monome**等）でしか役に立ちません。

機能：

- パターン内の全てのノート情報にアクセス
- パターン内での上下移動操作
- ページ切り替えによる、各トラックの表示
- フォロー表示：現在再生されているパターンの部分を自動追従
- 半音/オクターブ単位の転調機能
- 各ノート、又はベース・ノートのボリューム指定
- ノート情報のコピー&ペースト（ノート情報があるボタンを押したままにし、他のボタンを押す）

オプション：

- **orientation** - レイアウトの方向指定
 - [1:縦向き（vertical）、2:横向き（horizontal）]
 - **increment** - パターン内で上下移動する時にジャンプするライン数指定
 - [4, 6, 8, 10, 12]
-

Transport

Transportアプリケーションは**Renoise**での基本的なトランスポート機能（再生/停止等）を操作するものです。

機能：

- 再生/停止
- エディット・モードのオン/オフ
- パターン・ループ・ボタンのオン/オフ
- ブロック・ループ機能のオン/オフ
- パターン間の移動、又は移動予約の操作

オプション：

- **pattern_switch** - パターン間を移動した時の動作設定
 - [1:即座に切り替わる、2:切り替え予約]
- **pattern_play** - 再生ボタンを2回押した時の動作設定
 - [1:リトリガー、2:再生予約、3:停止]
- **pattern_stop** - 停止ボタンを2回押した時の動作設定
 - [1:パニック全音停止、2:曲の先頭へ戻る]
- **jump_mode** - 進む/戻るボタンの動作設定
 - [1:現在のパターンを移動させる、2:ブロック・ループを移動させる]

Navigator

Navigatorアプリはパターン再生とブロックループのサイズ/位置を操作する為のものです。それは**Matrix**アプリでパターン・トリガーするのと少し似ています。しかしパターンシーケンスの代わりに、ブロックループをセクションとして扱います。

機能：

- ブロックループが設定されていない状態でボタンが押された時、再生位置は即座に移動します。
- ブロックループが設定されている状態でボタンが押された時、ブロックループ位置は即座に移動します。
- 複数のボタンが押された時、ブロックループの範囲が設定されます。（ブロックループの範囲や位置は、押されたボタンの範囲や位置とほぼ同じ状態に設定されます。）
- ブロックループが設定されている状態でボタンを押したままにした時（ホールド状態）、設定されたブロックループは無視され、そのパターンを一通り再生します。

注意：

- **Navigator**アプリは常に現在表示されているパターン上で動作します。つまり、もし**Renoise**のパターンフォロー・ボタンをオフにして再生している場合、**Navigator**でポジションを変更すると現在表示中のパターン上に再生カーソルが戻ってしまいます。この動作を避けたい場合は、**Renoise**のパターンフォロー・ボタンをオンにしてください。
- ちょうど**Renoise**のブロック・ループのように、ナビゲーターは、1、2、4のような2の倍数の範囲において最良に働きます。いくつかの範囲と位置のコンビネーションが、他のものよりうまく具合に動作するのはこれが理由です。
- **Navigator**はテンポが崩れる事を避けようとします。その為に時々、前のパターンの最後にジャンプする事もあります。例えば、パターンのライン00に戻ろうとする場合に、実際はライン00よりも少し前に戻る必要があるからです。

オプション：

- このアプリにオプションはありません。

4. コントロール・マップの説明

コントロール・マップは、デバイスのパラメーターや物理的なレイアウトに関して書き記したXMLファイルです。**Duplex**とデバイスを繋げて何か出来るようにするにはコントロール・マップが必要です。ですから、もしあなたが自分のコントローラー用に改造したコントロール・マップを作りたいなら、少しその内容を知っておくといいでしょう。

基本的な例

```
<Device>
  <Name>Device name</Name>
  <Author>Name of author</Author>
  <Description>Here goes a description...</Description>
  <Parameters>
    <Group name="My Group">
      <Param minimum="0" maximum="64" type="button" name="First" value="CC#10"
/>
      <Param minimum="0" maximum="127" type="button" name="Second" value="C-4"
/>
      <Param minimum="0" maximum="127" type="fader" name="Third"
value="C-4|10" />
    </Group>
  </Parameters>
</Device>
```

ここでは沢山説明すべき事があります。最初から見てみましょう。

<Device> は全てのルート・ノードで、**<Name>**, **<Author>**, **<Description>** というセクションを含んでいます。これらの情報は、**Duplex**ブラウザのコントローラー画面の下に表示されます。

<Parameters> ノードは、実際のデバイスが記述されている場所です。その中の **<Group>** ノードには **<Param>** ノードが含まれています。私達は常にこのようにパラメーターをグループ化する必要があります。個別のパラメーターは、そのグループ名や、グループ内のポジション (**index**) によって特定されるからです。

いくつかの重要事項があります。

まず1つ目は、**<Param>** ノードは必ず **<Group>** ノードの中に置く必要があります。2つ目は、各パラメーターはその親グループの名前で特定されるので、**<Group>** には必ず名前を付ける必要があります。(各パラメーターに関しては必ずしもその必要はありませんが、例えばボタンに "Trigger 3" のような名前を付けておいた方がわかりやすいでしょう)

レイアウトの調整: **column** と **row**

コントロール・マップのレイアウトを調整したい場合、私たちは構造的なタグを持っています。しかしその前に、**<Group>** ノードに関してもう少し説明が必要です。**<Group>** ノードは、**@orientation** 属性を使ってレイアウト方向(縦向き・横向き)を指定する事が出来ます。こうする事で、各コンテナー(**<Group>**内の**<Param>**ノード)はその方向性に従います。縦向き(**vertical**)なら上から下、横向き(**horizontal**)なら左から右になります。

しかし、グループの内部を調整するだけでは不十分です。各グループをさらに**column**や**row**で囲って編成出来る必要があります。その為に **<Column>** と **<Row>** というタグがあります。これらのタグは

属性を持っていないので簡単に使えます。どんなコンテンツでも **<Column>** か **<Row>** ノードの中に置けばいいですし、他の **<Column>** や **<Row>** ノードも含める事が出来ます。ただし、きっちりとしたXML形式にする為に、タグを作ったら必ず閉じるようにしてください。（Duplexは、デバイス・プリセットを読み込む度にコントロール・マップを実証するので、ノードを閉じ忘れたり属性が無かったりするとエラー・メッセージを表示してくれます。それを読めば、どこを直せばいいかわかると思います）

パラメーター属性

続いて、パラメーター属性の説明です。既に述べたとおり、各パラメーターにはあなたの好きな名前を付けられますが、その名前は必ず必要な属性ではなく、単にコントロール・マップを見やすくするだけのものです。その一方で、**minimum** と **maximum** という属性は必ず指定する必要があります。それらはパラメーターの動作範囲（最小値/最大値）を記述しているからです。

@minimum, @maximum

基本的に、ほとんどのコントローラーはコミュニケーションする数値の範囲を持っています。ほとんどの（しかし、全てではありませんが）MIDIコントローラーは7ビットの範囲に限定されています。例えばフィルターをコントロールする為にノブを回した時、そのノブは**0-127**（7ビット）の間の数値を送信するでしょう。しかし時々、それとは違った数値で動作するパラメーターもあります。例えば、上記の例の最初のパラメーターでは最大値が**"64"**になっています。このように**minimum**と**maximum**の数値はお使いのデバイスによって違います。もしその数値にあまり自信が無かったら、お使いのコントローラーのMIDI implementationチャートを調べてみてください。

もう1つ、**min/max**値に関して重要な事があります。もしDuplexでボタンを作った場合、Duplexは自動的にその最小値/最大値をスイッチオン/オフの状態として使います。

@value

もう1つの重要な属性として **value** があります。コントローラーからメッセージを受け取る時にはいつでも、**value**を最初に見るでしょう。MIDIコントローラーの場合、その特別な表記法のおかげで、私たちが扱っているメッセージの「種類」を伝えることができます。OSCメッセージの場合は元々もっと一般的な表記なので、MIDIメッセージよりも簡単に解釈出来るでしょう。

Example	Message type	Matched because
CC#2	MIDI Control-change	Starts with "CC"
CC#7 6	MIDI Control-change + channel	Starts with "CC"
C-4	MIDI Note	Has "-" or "#" as second character
G#5 4	MIDI Note + channel	Has "-" or "#" as second character
PB	MIDI PitchBend	Starts with "PB"
PB 3	MIDI PitchBend + channel	Starts with "PB"
/led 3 2 %i	OSC message*	Starts with a slash

* OSCメッセージは文字列を直接的に埋め込む事が出来ます。「%i」は整数、「%f」は浮動小数点の数値を意味します。

@action

action属性は、OSCコントローラーでしか使われません。さらにそのコントローラーが、送信した時とは異なった形式のメッセージを受信する必要がある場合（monomeがそういうコントローラーなんだそうです）に限り、action属性を使います。その場合、action属性は数値としてコントローラー側に送られ、コントローラー側からはvalue属性がメッセージとして送られます。

@type

type属性もまた重要です。それは私達が今扱っている操作の種類をDuplexに伝え、バーチャル操作画面でどのようにその操作を再現するかを決定します。（注）buttonやtogglebutton等は、Duplexの内蔵のUIComponentタイプとは関係ありません。

Attribute name	Looks like	Additional arguments	Description
button			標準的な双方向のボタン。押した時（プレス）、離れた時（リリース）に数値を送信しますが、その内部の状態を変化させる事はありません。例えば、Launchpadやmonomeのボタンです。
togglebutton			内部の状態を切り替える双方向のボタン。このタイプは、リリースやホールドのイベントはサポートしません。例えば、BCF/BCRのボタンです。
pushbutton			プレス&リリースの数値を送信し、内部の状態も変化させる双方向のボタン。例としては、Automapの"momentary" ボタン、TouchOSCのプッシュ・ボタンです。
fader		@orientation	手動フェーダー
dial			基本的なロータリー・エンコーダー

@size

size属性は全てのパラメーターに使えます。それはバーチャル操作画面のパラメーター表示の相対的なサイズを決定します。もし"2"と指定すると、初期サイズの2倍くらいのサイズになります。また、"1.33"のような小数点の値を指定して、パラメーターのサイズを微調整する事も出来ます。

@orientation

orientation属性は、フェーダーにしか使いません。縦向きか横向きの指定です。