Duplex : MIDI/OSC controller framework for Renoise

This manual is valid for Duplex v0.98

目次

第１章：イントロダクション

Duplex って何？

Duplex は、幅広いハードウェアをプラグ＆プレイで扱う事ができ、Renoise の様々な機能をコントロールする事が出来るスクリプト・ツールです。Duplex に含まれるアプリケーションは、Renoise の基本機能を操るもの（ミキサー、DSPチェイン、パターン・マトリックス等）から、冒険的パフォーマンス・ツール（ステップ・シーケンサー等）まで、幅広く揃っています。

Duplex の最も秀でた特徴は、「双方向コミュニケーション」にあります。実際使えばわかりますが、あなたのコントローラー上の表示は常に Renoise の動作に同期します。これは Renoise のネイティブなMIDIマッピングだけでは出来ない事です。

スクリプトをプログラミング出来る人なら、Duplex フレームワークを使って、ハードウェアに幅広く対応可能なプログラムを作成する事も可能です。スライダーやボタンを作ったり、コントローラーの LED ライトの設定を標準的なRGB値で設定したり、送信する信号を自動的に最適化したりと、様々な作業を単純化して実現させてくれます。

Duplex は標準的な Renoise ツールですので tools.renoise.com からダウンロードでき、Renoise の画面上にドラッグ＆ドロップする事で簡単にインストール出来ます。Duplex は Renoise API を使って書かれているプログラムなので、誰でも拡張や改造が可能です。しかし、公式版は Renoise チームによって管理されています。

想定される対象者

このマニュアルの序盤は平易な用語で解説し、徐々に技術的な解説になります。もう少し整理して言うと、手に入れられる情報は"経験度"によって違う、という事です。

このマニュアルは本来、Duplex を全く知らなかった人（ビギナー用）、そして自分で改造したりカスタマイズしたい人（エキスパート用）の為に作られています。しかし経験度に関わらず、このマニュアルは基本を理解する手助けになるでしょう。ただし、Renoise フォーラムで得られる実際のアドバイスには敵わないかもしれません。

開発のロードマップ

最初 Duplex は Renoise 2.6 でのスクリプト対応と共に生み出され、それ以来の度重なるアップデートにより、新機能やアプリケーション、対応コントローラーが追加されてきました。全ての改良リストは [こちら] を見てください。

どんな機能がいまだ未実装かを調べる前に、ここに Duplex フレームワークがまだ対応していない機能（しかし将来予定されている機能）のリストを作りました。

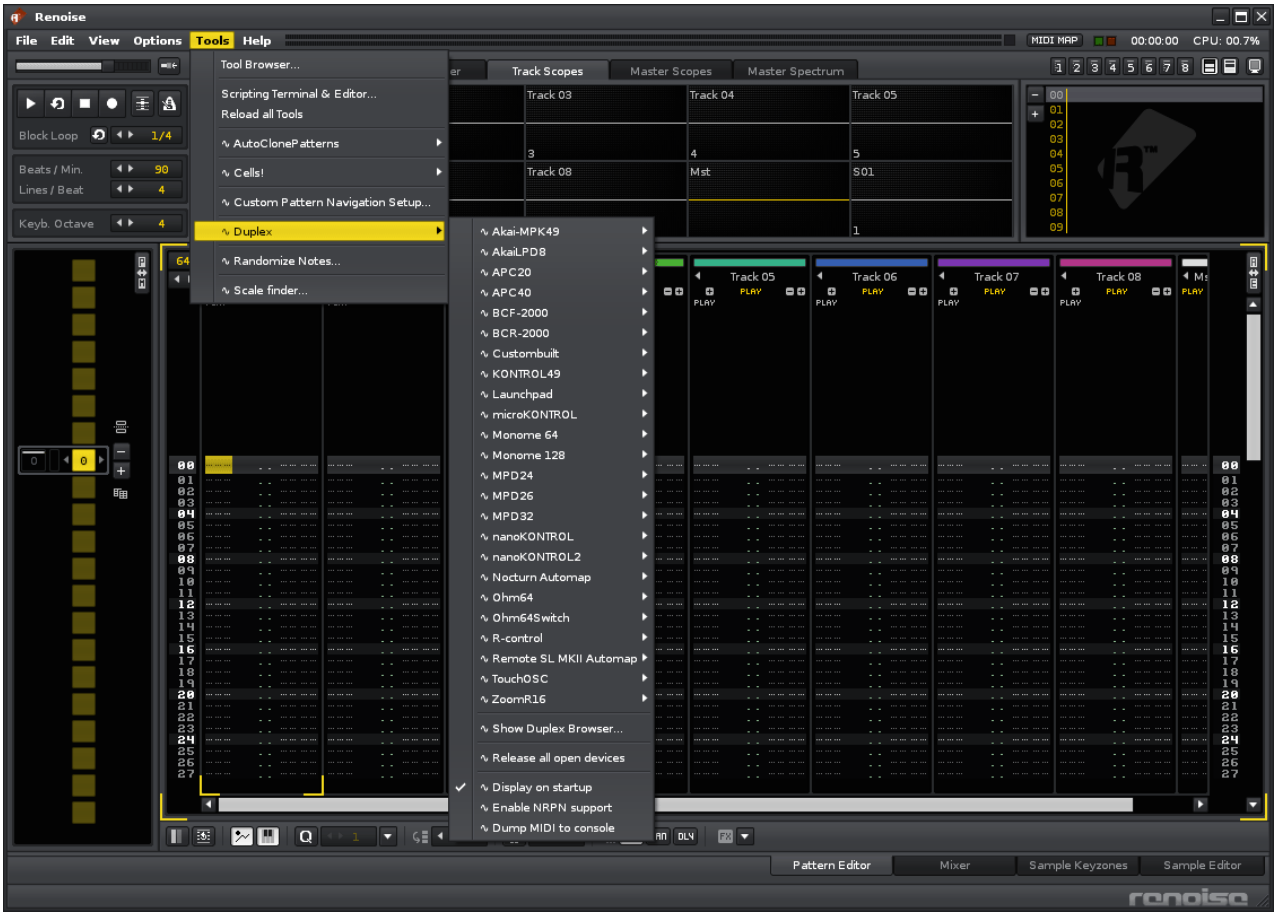
Description	Planned	Renoise API support (v3)
相対的エンコーダー（エンドレス・ノブ）の対応	Yes	Yes
リアルタイム・ノート・トリガリング	Yes	No
MIDIレンジの拡張（sysex による 7 ビット以上の信号）	Yes	Yes
コントローラー側のLCD画面のテキスト表示の更新	Yes	Yes

第２章：フロント・エンド（操作画面）

ほとんどの部分について、Duplex はそれ自身で説明を表示します。インターフェイスの各部にマウスをもっていくと、そのボタンやスライダーがどういう働きをするかがツールチップ内に表示されます。以下の説明は最も重要なユーザー・インターフェイス画面の簡単な説明です。

ツール・メニュー

Duplex をインストールすると、Renoise のメインメニューの [Tools] 内に "Duplex" が表示されるようになります。



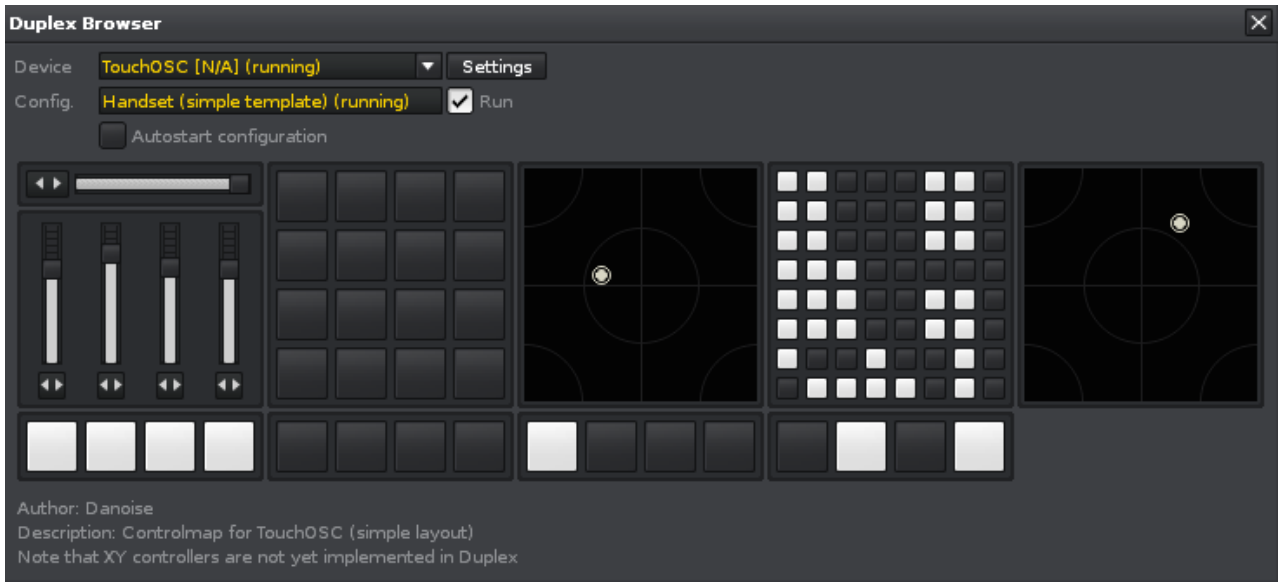
Show browser	Duplex ブラウザ（各コントローラー画面を表示・案内するための最初のUI）を表示します。
Display on startup	ここを有効にすると、Duplex ブラウザが Renoise の起動と同時に表示されます。ただし、１つ又は複数のデバイス・コンフィグレーションで "autostart" にチェックを入れておかないと実際にUI画面は表示されません。"
Release all...	起動中のデバイス・コンフィグレーションを全て閉じます。
Dump MIDI...	この機能は、うまく動作しないMIDIデバイスのデバッキングに役立ちます。

さらに、Duplex のメニューは、全てのプリセット (デバイス・コンフィグレーション) に素早くアクセス出来るようになっています。各コンフィグレーションはコントローラー毎のカテゴリに分けられ、それ自体の機能を説明するような名前 ("Launchpad Mixer + Matrix"、"Simple TouchOSC template"、等) で表示されます。これらのコンフィグレーションは、特に便利なアプリの組み合わせや、時には人気投票によって選ばれたものです。どれか 1 つを選択すれば、実際のコントローラーを模したUI画面が表示されます。

ブラウザ画面

Duplex ブラウザ上では、デバイス・コンフィグレーションのオン/オフや、各コントローラーと各コンフィグレーション間の切り替えが可能です。この画面には Duplex ならではの機能も含まれています。Renoise の UI コンポーネントを使って実際のコントローラーのレイアウトを模した "バーチャル・コントローラー" が表示されます。

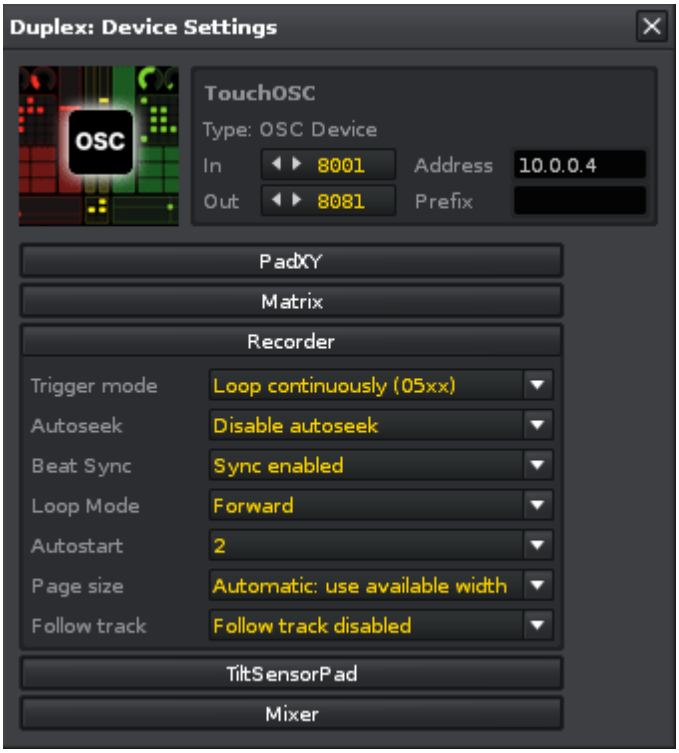
このブラウザ画面は完全なマルチ・タスク環境です。ほとんどのデバイス・コンフィグレーションは複数のアプリを含んでいますし、複数のコントローラーを同時に動作させる事も可能です。デバイスが動作中の場合、その表示名の後ろに "(rnuning)" という文字が現れます。



Run	そのアプリが現在アクティブで入力信号に反応する状態である事を示します。
Autostart	Renoise の起動時に、選択中のアプリを起動させるかどうかを設定します。
Options	デバイス・コンフィグのオプション画面を表示します。

オプション・セッティング画面

各デバイス・コンフィグレーションにはオプション・セッティング画面があります。そこにはデバイスのIN/OUTポート設定と、アプリ毎に特化したオプション設定があります。これらの設定は（IN/OUTポート設定も含めて）デバイス・コンフィグレーション毎に独立しているので、例えばコンフィグ毎に別々のオプション設定を施されたミキサー・アプリを持つ事が可能です。そのオプションは一旦設定すれば永続的に記憶されるので、Duplex 自体を再インストールするまで保存されます。また、オプション設定の変更はリアルタイムで効果が反映されます。



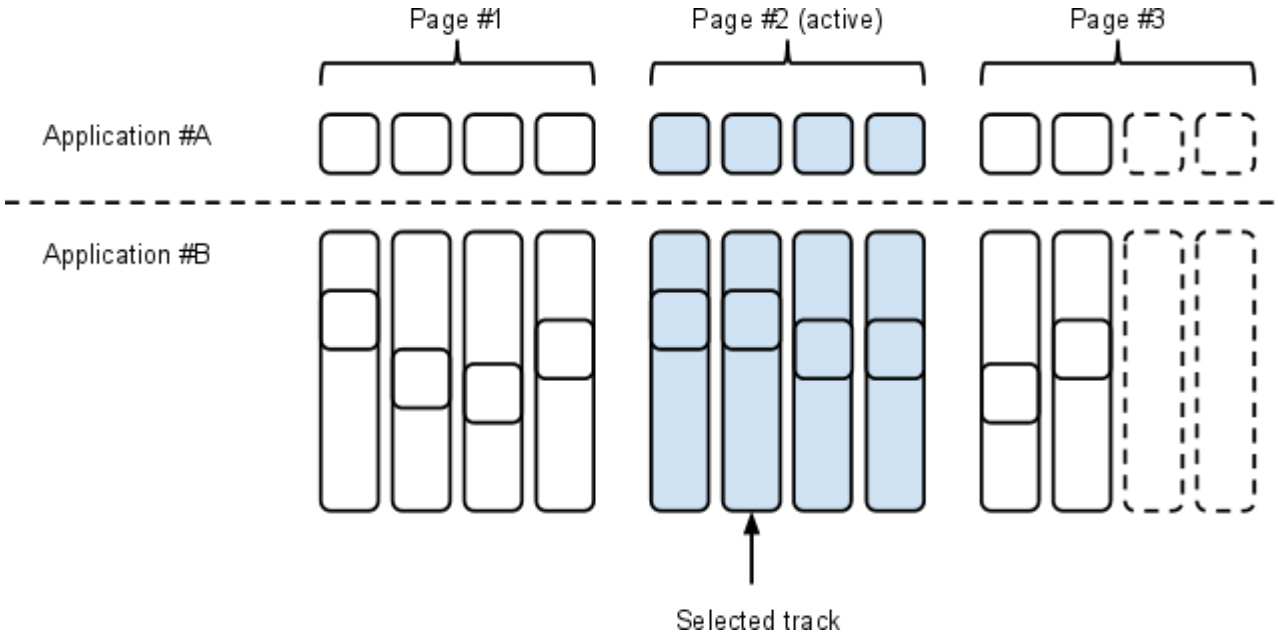
第 3 章：同梱されているアプリケーション

Duplex において "アプリケーション" とは、特定のタスクを実行し、外部コントローラーから操作する事が可能なコードの 1 つです。アプリケーション自体にはユーザー・インターフェイスはありません。あなたのコントローラーがインターフェイスです。

共通機能：ページ・ナビゲーション

ほとんどのアプリはお互い機能的に統一感があるようにデザインされています。そういった手法の一つに "ページ・ナビゲーション" があります。基本的に、各アプリは現在のトラック/パターン/デバイス等から情報を活発に収集し、その情報を元に動作します。大抵の場合、トラック数やパターン数を同じサイズの区分（つまり "ページ"）に分ける為の情報を集め、現在のポジションが変わった時に画面を更新します。

下図は、ページ・サイズが 4 に設定されている時、どのように10トラックが 3 つのグループに区分けされるかを表しています（当然、アプリ#Aと#Bは共に同じページ・サイズが設定されていないとうまく動作しません）。



ページ・ナビゲーションは Renoise のアクティブ・パートに追従しなくても動作する事が出来ます。この事はページ・ナビゲーションをシンプルな「前/次のページ」という形式にしてくれます。あなたがコントローラーを使い、友人が Renoise を使う場合なんかに便利です（二人ともが Renoise のアクティブ・パートを取り合ったら混乱してしまいます）。

ページ・ナビゲーションに対応したアプリは、必ず以下の 2 つのオプションを備えています：

- Follow_pos (or follow_track, follow_pattern): Renoise上 の現在カーソルがある部分を追跡します。
- Page size: ページのサイズを指定します。

各アプリケーションの紹介

では、あなたが目的にあったアプリを見つけられるように、各アプリを簡単に解説しましょう。ですが、以下の情報は完璧ではありません。もし、もっと詳細な各アプリの使い方を知りたければ、表中のリンク先を見てください。実際のアプリのソースコードには、あなたのカスタム・マッピングを作るのに必要な情報が全て記されています。

Name	Description	Links
Matrix	「マトリックス」アプリは、パターン・マトリックス画面とパターン・シーケンス画面をコントロールします。マトリックス画面上の各スロットのミュート状態をオン/オフし、パターン・シーケンス画面上の任意のパターンをトリガーしたりループさせたり出来ます。ページ・ナビゲーション対応。	Source
Mixer	「ミキサー」アプリは、Renoise のミキサー画面をコントロールします。自動的に現在有効な全てのトラック数を認識し、ボリューム、パン、ミュート状態等の複数トラックに渡る情報を並べて表示させる事が出来ます。ページ・ナビゲーション対応。	Source
Effect	「エフェクト」アプリは、DSPチェイン内の全てのパラメーター・スライダーをコントロールします。現在選択中のデバイス（エフェクター）を追跡し、そのデバイスが持っている全てのパラメーターをページ・ナビゲーション・システムを使って表示させる事が出来ます。 ３種類のパラメーター表示オプションがあります： オートメーションされたパラメーターだけ表示（各トラック毎）、ミキサー画面上に表示されているパラメーターだけ表示（各トラック毎）、全てのパラメーターを表示（各デバイス毎）。	Source
StepSequencer	「ステップ・シーケンサー」アプリは、あなたのコントローラーを使って、パターン・エディター上に音名とボリューム・データをリアルタイムに入力する事を可能にします。単純なコピー＆ペースト機能も備えています。	Forum Source
Transport	「トランスポート」アプリは、Renoise の基本的なトランスポート機能をコントロールします。再生/停止（１ボタンでの切り替え、又は２ボタンに割り振り）、前/次のパターンへ移動、エディット・モード、ループ、ブロック・ループ、メトロノーム等の各スイッチの管理。	Source
Navigator	「ナビゲーター」アプリは、現在のパターン上の再生位置を光の移動によって表示しコントロールします。ブロック・ループのサイズや位置の設定も可能です。	Source

TrackSelector	「トラック・セレクター」アプリは、トラック間の移動をページ・ナビゲーション・システムを使ってコントロールします。さらに、2ボタン(前/次のトラック)、ノブ又はスライダー(トラック間をスクロール)でのコントロールと、3つのダイレクト・ジャンプ(第一トラック、マスタートラック、センド・トラックへのジャンプ)も設定可能。	Source
Metronome	この「メトロノーム」アプリだけは、見本的な存在です。双方向アプリケーションの基本的なプログラミングを学びやすいようにコードが記述されています。実際にメトロノーム機能を操作したい場合はトランスポート・アプリを使ってください。	Source
Recorder	「レコーダー」アプリは、あなたの声やギター等、Renoiseに流れ込んだオーディオ信号を録音する事が出来る特別なアプリです。これはライブ・パフォーマンス・ツールとしてデザインされています。 まず、トラック・ボタンを押すと録音パネルが表示され、もう一度押すと録音が始まります。録音が完了すると、その結果が自動的にパターンに貼り付けられ、(オプションとして)ループや同期設定も可能です。録音ファイルは数通りストックしておくことが出来、それらを瞬時に入れ替える事も出来ます。また、違う長さの録音ファイルをループさせるオプション(ポリリズム)もあり、それらの録音ファイルはRenoiseの曲毎に保存されます。ページ・ナビゲーション対応。	Manual Source
NotesOnWheels	「ノーツ・オン・ホイールズ(N.O.W)」アプリは、ステップ・シーケンサーとフレーズ・アルペジエーター機能を持ったパフォーマンス・ツールです。1~12ステップのシーケンスを作成可能で、個別の要素(音名、ベロシティ、サンプル・オフセット等)をリアルタイムにコントロール出来ます。特徴として、それ自身が出力したデータを読み取る事が可能で、それによって既にパターン上に入力されたシーケンスを"拾い上げる"事が出来ます。 データの入力は、幾つかのボタンと"ノブ又はフェーダー"があるコントローラー(ここが重要です)によって、自由自在に行えます。さらに、音名データ入力と移調は、パソコン・キーボードや外部MIDIキーボードを使っても入力する事が出来ます。	Forum Manual Source
SwitchConfiguration	「スイッチ・コンフィグレーション」アプリは、あなたのコントローラーが数種類のコンフィグを持っている場合、それらの切り替えを可能にします(2ボタン。前/次のコンフィグへ切り替え)。ただし、全てのコンフィグ内(出来れば同じボタン)にこの機能を割り当てておかないと、コンフィグの切り替えが一方通行になって(戻れなくなって)しまいます。	Source

GridPie	「グリッド・パイ」アプリは、複数のパターンで構成された曲の任意のパーツを組み合わせて即興的に鳴らす事が出来るリアルタイム・パフォーマンス・ツールです。このアプリを起動するとパターン・マトリックス画面の全スロットがミュート状態になり、任意のスロットを選択する事で、一番下にあるグリッド・パイ用のパターンにデータがコピーされループ再生される仕組みです。その為、このアプリもページ・ナビゲーションに対応しています。 右のフォーラムへのリンクは、オリジナルのグリッド・パイ・ツールです。その後、Duplex のアプリとして移植されました。	Forum Source
XYPad	「XY パッド」アプリは、Renoise 内蔵の *XYPad デバイスをコントロール出来ます。MIDIノブ/ボタン、タッチパッド、マルチ・バリュー OSC メッセージ（加速度センサー等）によって操作する事が可能です。	Forum Source
Rotate	「ローテート」アプリは、音データと（オプションとして）オートメーション・データの位置を上下に循環させる事が出来ます。トラック毎、又はパターン全体のデータを動かす事が可能です。	Source

第 4 章：コントローラーへの対応

あなたのコントローラーを Duplex で使う

とにかくまず、あなたのコントローラーが既にサポートされているかどうか [リストを確認] してみてください。Duplex が対応するコントローラーは増え続けています（現在24個）。各コントローラーには少なくとも 1 つ以上のコンフィグレーションが与えられています。もしあなたのコントローラーがサポートされていない場合、まずそのコントローラーがどういった用途に向いているか考えるといいでしょう。例えば、ノブが沢山付いたコントローラーなら、エフェクトやボリュームを操作出来ますし、ボタンが沢山並んだグリッド・コントローラーなら、マトリックス画面やシーケンサー等の操作に向いているでしょう。

新規のコントローラーへの対応

新しいコントローラーに対応する為には、いくつかの調査が必要です。あなたの経験度にもよりますが、自分でやってみたい、又は誰かに助けて欲しい、と思うかもしれません。どちらにせよ以下に、標準的な MIDI/OSC コントローラーのセットアップ手順を記述します。

最初に、そのコントローラーに関する情報をとにかく出来る限り集める事が大切です。大抵の開発メーカーは、詳細な MIDI インプレメンテーション・チャートや OSC プロトコルの仕様をマニュアル等に記載しています。これらは、コントローラーと接続しデータを受け取る為に必要です。少なくとも私たちは、Duplex 側からコントローラーへ送信するべき数値がどういったものなのかを教えてくれる資料が必要です。逆にコントローラー側から Duplex へと送られる数値は MIDI-OX 等のユーティリティを使えば簡単に調べられます。Duplex の [対応コントローラー・リスト] には、そういった仕様書の典型的な例へのリンクがいくつかあります。

そしてこういった知識を得たなら、次は適切なコントロール・マップを作成しましょう。その為には、そのコントローラーがどういった種類のボタンを持っているかを知り、入力タイプを決める必要があります（１つのボタンのタイプがわかれば、恐らくそのコントローラー内のボタンはほとんど全て同じタイプでしょう）。例えば、押さえた時にオンとオフが切り替わるタイプの LED ボタンなら、このボタンには "togglebutton" とラベル付け出来るで

しょう。

一旦タイプが決定されれば、Duplex はこのボタンを様々なタイプのボタンとして扱う事が出来ます。一度押さえると点滅し始めたり（レコーダー・アプリのボタン）、押さえている間だけ光って放すと消えたり・・・これが双方向コミュニケーションの恩恵です。

アプリケーションは最適な方法でコントローラーに使われます。それゆえに、他に一切の音楽ソフトを起動していない状態でコントローラーを接続し、まずボタンのタイプを確定する事が重要です。私たちはそのボタン自体の "純粋な動作タイプ" が知りたいのです。それがわかれば、有効な入力タイプ（コントロール・マップの章の "@type" 欄を参照）の中から当てはまるものを選んで、コントロール・マップにラベリングする事が出来るのです。

現存する全てのコントローラーに当てはまる「定番のアプローチ」というのはありません。新しいコントローラーに対応する事は、複雑な仕事となるでしょう。もしかしたら特別な初期化コードが必要かもしれませんし、ドライバのようなものが必要になる場合もあるかもしれません。そういった個別の問題については、このマニュアルでは解説しませんが、それに関しては Renoise フォーラムでアドバイスを受ける事が出来るでしょう。

コントローラーについてのトラブル・シューティング

一般的に、可能な場合はいつでも、操作はプラグ＆プレイであるべきです。以下は、Renoise フォーラムにトピックを立てる前に、ご自身でチェックして欲しい事柄です。

1. MIDI コントローラーの場合、Duplex のオプション・セッティング・パネルで正しい In/Out ポートが選択されているか確認してください。*OSC* コントローラーの場合、ネットワーク・コミュニケーションが有効かどうか、そしてファイヤー・ウォールによってブロックされていないか確認してください。

2. デバイス・コンフィグレーションの 1 つを開き、バーチャル・コントローラー画面下部のコメントを読んでください。特別なプリセットや エディター用ファイルが必要な場合はそこに注意書きがしてあります。

3. Duplex メニューの "MIDI dump" 機能を有効にすると、Renoise のスクリプト・ターミナル画面に、現在送受信されている信号が全て表示されます。これはデバッギングに役立ち、セットアップの中のエラーを見つける手助けになるかもしれません。

第 5 章：コントロール・マップの仕組み

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ブラウザー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 <Column> と <Row> を持っています。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> ノードでさえも、そこに置く事が出来ます。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))

コントロール・マップを複数のグループに分割する方法

あなたのデバイス・コンフィグレーションをカスタマイズしたい場合、コントロール・マップを複数のグループに分割する必要があるかもしれません。このような例として、LaunchPad の "Matrix + Mixer + Transport" コンフィグを見てください。そこでは LaunchPad の画面を分割し、2 つのアプリが上下に表示出来るように工夫されています。

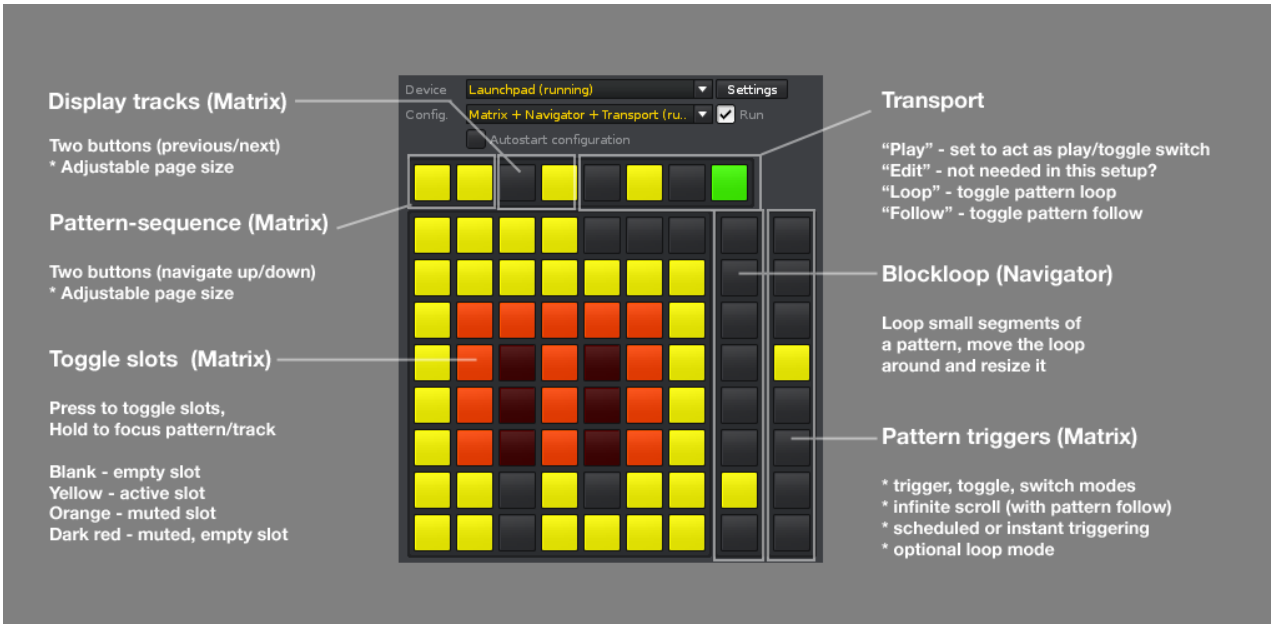
基本的に、外部エディターでコントロール・マップを開き、<param> ノード達のグループを新しいレイアウト用に再編成する必要があります。この為には <Column> や <Row> が使えます。再度 LaunchPad のコントロール・マップを見てもらえばわかりますが、大きな 1 つのグループを 2 つに分割した良い例です。

正しく編集出来るようになるには少し練習が必要ですが、コントロール・マップはロードの度に検証されるので、もし不適切な記述がある場合は Duplex が何らかのエラー・メッセージを表示してくれます。

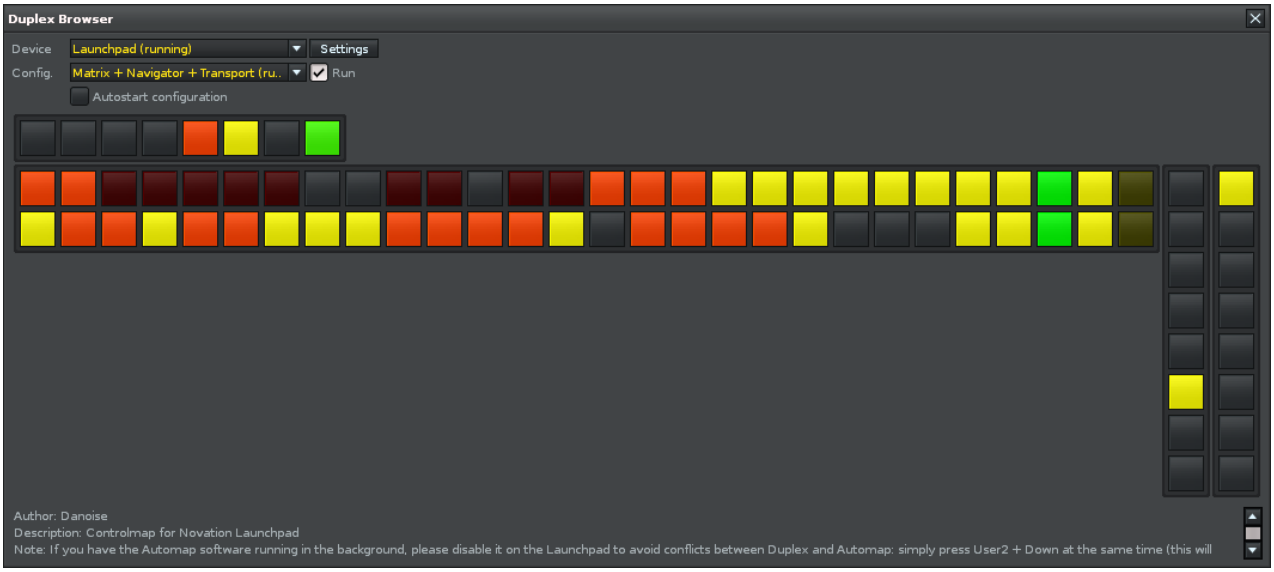
コントロール・マップのハッキング (あまり推奨しないテクニック)

一旦コントロール・マップの基本的な構文を理解すれば、何かそれを使って創造的なマップを作ってみたくなるかもしれません。コントロール・マップの目的はコントローラーの物理的なレイアウトを Duplex 内で再現する事にあるので、それを応用すれば、実際とは違うレイアウトのマップを作って Duplex を騙す事も可能です。

例えば、下の標準的な LaunchPad の Matrix コンフィグのレイアウトを少し弄ってみましょう。



メイン "Grid" グループの columns の数値を 8 から32に変更すると、下図のような全く違うレイアウトが出来ます。これを使えば32トラック、2パターンの構成でコントロール可能です（しかし、もはやこれは LaunchPad ではなくなってしまっていますが）。



コントロール・マップ・レファレンス：グループ属性

@orientation

vertical（縦並び）又は hrizontal（横並び）の方向性を指定します。これはコンテンツ（グループ内の <Param>ノード）の並び方を上から下（vertical）又は左から右（horizontal）に変更します。

@columns

あなたのコントローラーが沢山のボタンやノブを持っていて何列か並んだ状態で存在する場合、columns 属性を使います。この属性の数値はコラムの数（行数）を意味し、グループ内のコンテンツを「折り畳んで並べる」ことが出来ます。

@colorspace

いくつかのコントローラーはマルチカラーのLEDボタンを持っています。この colorspace 属性は、どの色が使用可能かを指定し、またそれはバーチャル・コントローラー画面上で表示される色の指定にもなります。

これは LaunchPad のカラースペースです。赤と緑が 4 レベル使用可能です。

```
colorspace = "4,4,0"
```

これは単色（モノトーン）仕様の monome のカラースペースです。

```
colorspace = "1,1,1"
```

また特殊な例として、色を解析してコントローラーが認識できる数値に変換するような、特別なコードが必要な場合もあります。もっと詳しくカラースペースについて知りたければ [Duplex API ドキュメンテーション] を見てください。

コントロール・マップ・レファレンス：パラメーター属性

@name

パラメーターに名前を付ける事が出来る属性です。これは単にコントロール・マップを読みやすくする為に使われるもので、必ず必要な属性ではありません。

@minimum, @maximum

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。一方で、フェーダーやノブはその間の数値も使います。

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を見ると、最大値が "64" になっています（これは Ohm64 の場合です）。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 インプレメンテーション・チャートを調べてみてください。

@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読出来るでしょう。

（訂正：以前の Duplex マニュアルでは、MIDI チャンネルを指定する "Ch" の説明が抜けていました）

Example	Message type	Matched because
CC#2	MIDI Control-change	Starts with "CC"
CC#7|Ch6	MIDI Control-change + channel	Starts with "CC"
C-4	MIDI Note	Has "-" or "#" as second character
G#5|Ch4	MIDI Note + channel	Has "-" or "#" as second character
C--3	MIDI Note (negative octave)	Has "-" or "#" as second character
PB	MIDI PitchBend	Starts with "PB"
PB|Ch3	MIDI PitchBend + channel	Starts with "PB"
/led 3 2 %i /xyz %f %f %f	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・コントローラー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。

Type	Looks like	Attributes	Description
button		@aspect	標準的な双方向のボタン。押した時（プレス）、離した時（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		@aspect	押した時に数値を送信し、内部の状態を切り替える（場合もある）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、Automapの "normal/toggle" ボタンです。

pushbutton		@aspect	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。
xypad		@invert_y @invert_x	XY パッドは、２つの数値を１つのペアとして扱う特別な操作端子です。TouchOSC の XYPad コンフィグを見てください。
fader		@orientation	手動フェーダー
dial			基本的なロタリー・フェーダー

@size

各パラメーターは @size 属性を設定可能です。バーチャル・コントローラー画面上のボタンやノブの相対的なサイズを決定します。例えば "2" と指定すれば、初期状態の２倍の大きさになります。"1.33" のような小数点の数値も扱えます。

@orientation

この属性は、フェーダーにしか使えません。縦向きか横向きの指定です。

@aspect

この属性は、ボタンにしか使えません。ボタンの縦横比（高さ）を指定します。例えば "0.5" なら、高さが幅の 50% になります。

@invert_x, invert_y

この属性は、XYPadにしか使えません。X軸/Y軸の数値を反転します（値は "true" か "false" しかありません）。

@skip_echo

基本的に、この属性を使うと、Duplex からコントローラーへ更新信号を送らなくなります（コントローラーが双方向に対応していない場合に使う）。

第 6 章：デバイス・コンフィグレーション

デバイス・コンフィグレーションとは、"コントロール・マップ"、"デバイス・クラス"、"アプリケーション"の全てを結びつけるファイルです。デバイス・コンフィグレーション自体は単純なテキスト形式のファイルで、Duplex がロードされる時に lua 解釈プログラムによって分析されます。ですから、実際それ自体はコードではありませんが、100%正しい構文で記述する必要があります。さもないと、エラー・メッセージが表示されてしまいます。

注）これら Duplex の各ファイルがどのように関わり合っているかを [簡単に説明した記事] があります（Renoise Gate の フォーラム）。そちらも参照してください。

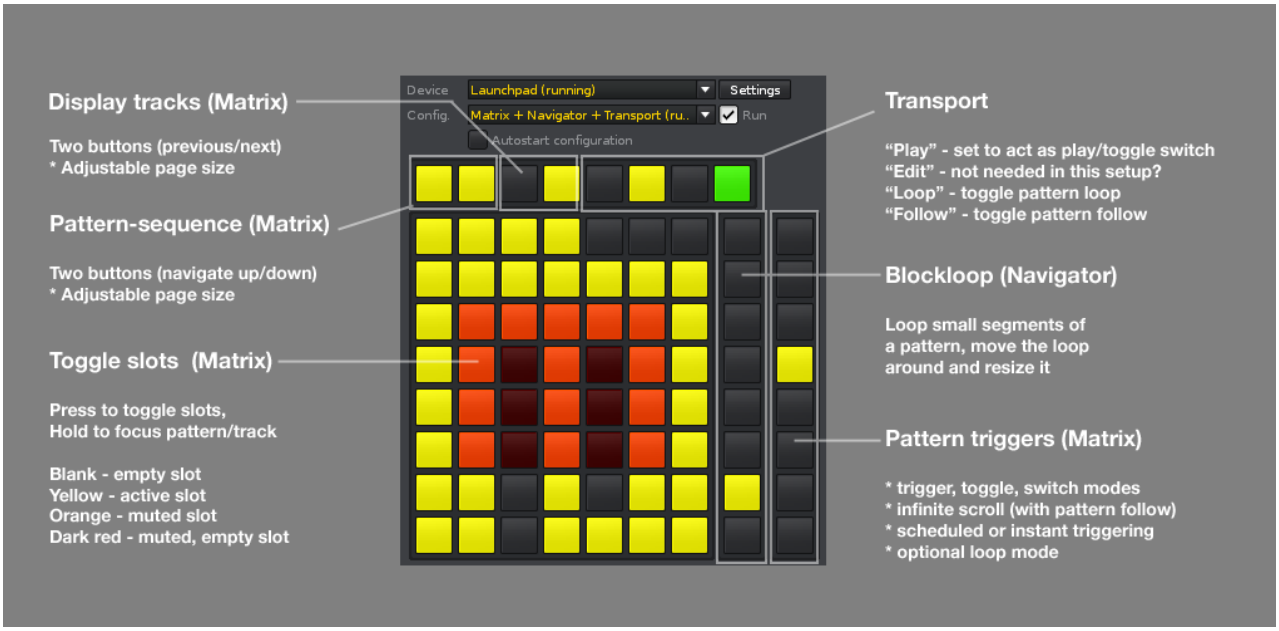
実際のデバイス・コンフィグレーションを使っての解説

この見本のコンフィグを使って学んでみましょう：

```
duplex_configurations:insert {
  name = "Matrix + Navigator + Transport",
  pinned = true,
  device = {
    class_name = "Launchpad",
    display_name = "Launchpad",
    device_port_in = "Launchpad",
    device_port_out = "Launchpad",
    control_map = "Controllers/Launchpad/Controlmaps/Launchpad_Matrix.xml",
    thumbnail = "Controllers/Launchpad/Launchpad.bmp",
    protocol = device_MIDI_PROTOCOL,
  },
  applications = {
    Matrix = {
      mappings = {
        matrix = {
          group_name = "Grid",
        },
        triggers = {
          group_name = "Triggers",
        },
        sequence = {
          group_name = "Controls",
          index = 1,
        },
        track = {
          group_name = "Controls",
          index = 3,
        }
      },
      options = {
        sequence_mode = 2,
      }
    },
    Navigator = {
      mappings = {
        blockpos = {
          group_name = "Navigator",
        }
      }
    },
    Transport = {
      mappings = {
        edit_mode = {
          group_name = "Controls",
```

```
        index = 5,
      },
      start_playback = {
        group_name = "Controls",
        index = 6,
      },
      loop_pattern = {
        group_name = "Controls",
        index = 7,
      },
      follow_player = {
        group_name= "Controls",
        index = 8,
      },
    },
    options = {
      pattern_play = 3,
    }
  },
 }
}
```

これが LaunchPad 上に表示される結果です。



見てわかるように、コントローラー画面上に配置出来る機能はある程度限られています（デバイス・コンフィグレーションの構文は下へ下へと継ぎ足す事が出来ますが）。それと幾つかの注意点があります：

- "name" 欄で使ったコンフィグ名は、２度は使えません（コントローラー毎に１つずつしか同じコンフィグ名は使えません）
- "pinned" 欄では、Tools メニュー内にそのデバイス・コンフィグを表示するかどうかを指定します。"false" を指定すると、Tools メニューには表示されなくなります。（ブラウザからアクセスする事は可能）
- それ以下のデバイス・プロパティ部分については、大体は見てわかると思います。ただし、*OSC* と MIDI デバイスでは違いがあるので、*OSC* コントローラー用のコンフィグを作りたい場合は、既に対応済みの *OSC* コントローラー（monome, TouchOSC 等）のコンフィグをコピーして改編してください。

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

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。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（ミキサーのマスター・ボリュームを表す "master" 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、もしテト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名を付けるでしょう。

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・グルー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はその碁盤の目の様な画面を１つのマッピングだけ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",
  },
  start_playback = {
       description = "Start playback",
  },
  loop_pattern = {
       description = "Toggle pattern looping",
  },
  [more mappings...]
}
```

いくつかのマッピングでは "orientation" プロパティも対応しています。この値は group_name や index と並んで指定されるもので、そのマッピングの方向性が縦向き（VERTICAL）か横向き（HORIZONTAL）かを決定するものです。この指定が必要かどうかはアプリケーション毎に違うので、各アプリのソースコード（“self.mappings” 欄）を確認してください。

デバイス・コンフィグレーション：アプリケーション・オプション

各アプリケーションは、マッピング・セットの他に、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。大抵の場合、それらのオプションは、Duplex ブラウザのオプション・セッティング・パネル上で簡単に変更出来るので、ここで必ずしも指定する必要はありません。しかし、オプション設定をデバイス・コンフィグ上に記載しておけば、その状態をデフォルトとして登録しておく事が出来ます。

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 "swich_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。もう少し下の "option constants" 欄を見れば、どのような機能が割り当てられているかわかります。マトリックス・トリガー・オプションは以下の種類があります：

```
self.SWITCH_MODE_STOP = 1
self.SWITCH_MODE_SWITCH = 2
self.SWITCH_MODE_TRIG = 3
self.SWITCH_MODE_SCHEDULE = 4
```

デバイス・コンフィグレーション：アプリケーション・パレット

アプリケーション・マッピングやオプションの他に、カスタム・パレットを指定する事が出来ます。パレットは、"カラー" と "テキスト" という 2 つの数値で構成されています。カラーはその名の通り、RGB カラー (0X00 から 0XFF) を指定します。もしコントローラーがカラー表示に対応していない場合、テキスト値がバーチャル・コントローラー画面上に表示されます。どちらの値が使用可能かはアプリケーションのソースコードを見てください。

以下は、XYPad アプリのデフォルト・パレットです。これがどのようにカスタマイズされているかは、[LaunchPad XYPad コンフィグ] を見てください。

```
self.palette = {
  foreground = {
    color={0xFF,0xFF,0x00},
    text="■",
  },
  background = {
    color={0x40,0x40,0x00},
    text="□",
  },
}
```

付録 A：よくある質問と答え

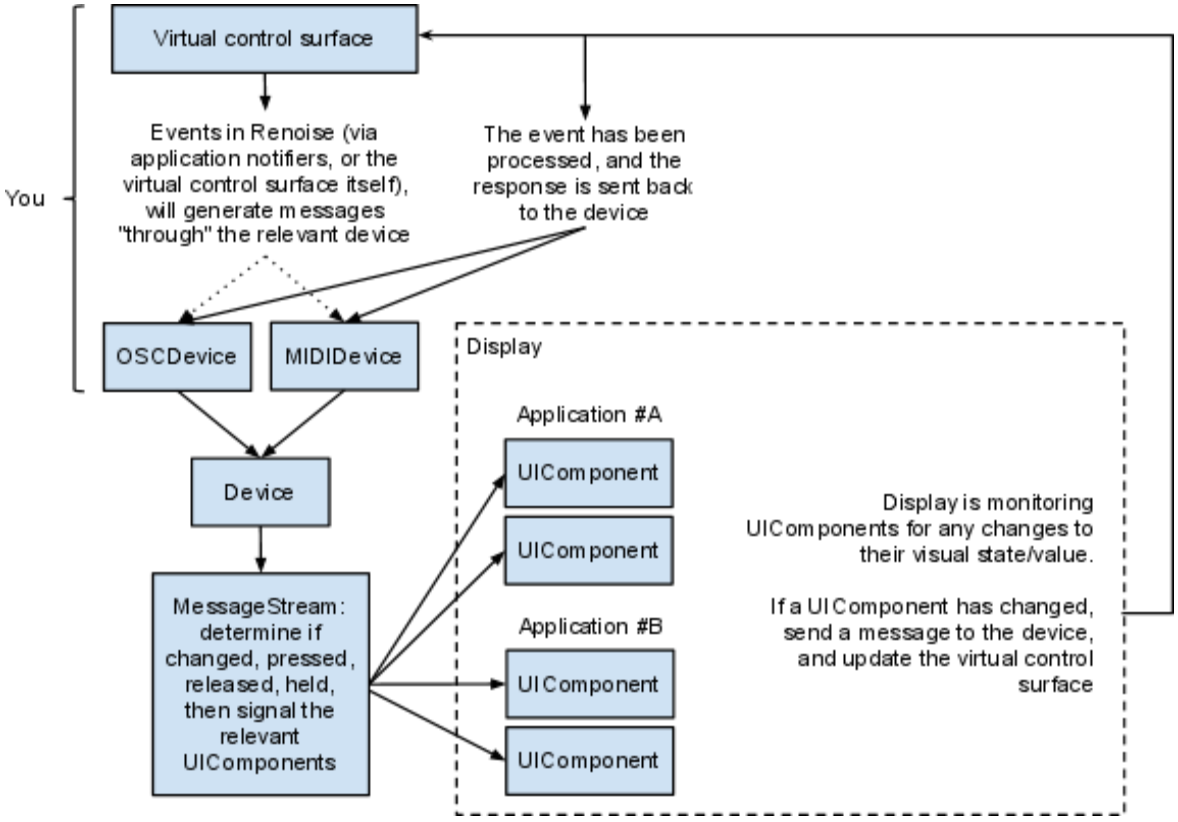
Question	Answer
私のコントローラーには対応していますか？	まず Duplex をインストールして、メニュー内のリストを確認してみてください。その中にあなたのコントローラーがあれば、答えは「YES」です。もしそこに無い場合は、[コントローラー・リスト] の方も確認してみてください。
私のコントローラーがうまく動作しません	[コントローラーについてのトラブルシューティング] を読んでみてください。
Duplex フォルダをどこに置けばいいですか？	Duplex フォルダは com.renoise.Duplex.xrnx という Renoise ツール・フォルダ内に置かれています。Renoise を起動した状態なら次の手順で開けます： メイン・メニューの [Tools] 欄の “Tool Browser” を表示させます。リスト内の Duplex を探してその上で右クリック・ メニューを開き、“Reveal in Explorer” というオプションを選びます。
アプリケーション・クラスはどのように置けばいいですか？	アプリケーション・クラスは Duplex/Applications というディレクトリ下に置かれています。
デバイス・コンフィグレーションはどのように置けばいいですか？	デバイス・コンフィグレーションは以下のディレクトリにあります。 Duplex/Controllers/[Device Name]/[Device Name]/Configurations/ 注) [Device Name] 部分をあなたのコントローラー名に置き換えてください。
コントロール・マップはどのように置けばいいですか？	コントロール・マップは以下のディレクトリにあります。 Duplex/Controllers/[Device Name]/[Device Name]/ControlMaps/ 注) [Device Name] 部分をあなたのコントローラー名に置き換えてください。 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の設定を初期状態に戻せますか？	Duplex のセッティングは Duplex フォルダ内の “preferences.xml” というファイルに保存されています。初期状態に戻したい場合は、Renoise が起動していない状態で、このファイルを削除してください。

私が気に入っていた設定が全て消えてしまいました	Duplex の新しいバージョンをインストールすると、全ての設定が初期状態に戻ってしまいます。ですから、"preferences.xml" やカスタマイズしたデバイス・コンフィグ、コントロール・マップ等は**必ずバックアップを取っておいてください。** preferences.xml を復元するには、Renoise が起動していない状態で、このファイルを上書きコピーしてください。
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	それはおそらく、そのMIDIコントローラーが Renoise 環境設定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 ブラウザー以外でデバイス設定する必要はありません。
Duplex を使ってノートをトリガーする事（音を鳴らす）は可能ですか？	「ノートをトリガーする」という意味にもよります。もし「パターン・エディターにノートデータを入力する」という意味なら、ステップ・シーケンサー等のアプリを使えば、それは可能です。 もし「パターン・エディターを通さずにインストゥルメントをプレイしたい」という意味なら、それは不可能です。現状の Renoise API にはそのようなリアルタイム処理は含まれていません。
私は２つの Launchpad をもっています。それらを並べて使う事は出来ますか？ **同じ種類のコントローラーを複数同時に使えますか？**	はい。デバイス・コンフィグレーションをコピーし、別の名前を付けて表示させる事が可能です。この方法で、同種のコントローラーを異なったポートに接続して、複数同時に使う事が出来ます。 そのコントローラーの使いたいデバイス・コンフィグを全てコピーした新規コンフィグ・ファイルを作ります。そしてそのファイルを開き "display_name" 欄を、例えば "Launchpad (2)" という風に変更します。これで、次回 Renoise 起動時には、コピーファイルが別のデバイスとして表示されるようになります。新しいデバイスの Input/Output ポートの設定を忘れないでください。

１つのコントローラーで複数の Renoise を操作する事は出来ますか？	２つの Renoise 内で同一のデバイスを使う事は、少しトリッキーですが可能です。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。
全てのコントローラーをオフにする方法はありますか？	Duplex ブラウザを開いてデバイス・リストの一番上の "none" を選択してください。これで現在接続中のコントローラーは全て解除されます。Duplex メニュー内の “release all devices” を選んでも同じです。
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	Duplex の終了時に、全てのライトをリセットするという機能はありません。しかし、そのコントローラーの次回起動時には全ての機能が初期化されるので、心配はいりません。
Renoise と MIDI controller 間の MIDI 信号のやりとりを見ることは出来ますか？	はい、Tools > Duplex > "dump MIDI" にチェックを入れてください。これで、Renoise スクリプト・ターミナルのコンソールに、現在送受信されている MIDI メッセージが表示されます。
個々のボタンやスライダーの位置を変更出来ますか？ **私のコントローラーでもっと多くのアプリを使えますか？**	はい。[第５章：デバイス・コンフィグレーションの仕組み] を見てください。
Renoise スクリプト・エディター＆ターミナルを有効にする方法	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。

付録 B: 技術的情報

これは Duplex 内の信号の流れを表した図です。



バーチャル・コントローラー画面（Duplex ブラウザ上のボタンやノブをクリックした時）や、実際の OSC/MIDI コントローラー（物理的なボタンやノブを触った時）のどちらを触っても、新たな信号が発生します。

アプリケーションを通してメッセージが処理され画面表示が更新された時に、信号が Duplex から離れます。そしてその時、操作が変更された事が決定されます。

付録 C: 役立つリンクや資料

1. Duplex ベータ版のスレッド (Renoise forum) 常にここに最新版がアップされています
2. Duplex の対応コントローラー・リスト, 既に対応済み、又は対応作業中のコントローラー・リスト表
3. Duplexファイルの編集方法 (Renoise-Gate 掲示板) コントロール・マップやデバイス・コンフィグレーションの編集方法の解説。
4. 新しい Duplex アプリケーションの作り方 , "in-depth" コーナーでの最初の記事 (Renoise forum)
5. プログラミング上級者なら、利用可能なクラスやそのプロパティと方法について学ぶ為に、Duplex API docs を読んでみてください