

USER MANUAL



INDEX

1 - INTRODUCTION.....	5
2 - DEVELOPMENT ENVIRONMENT.....	6
2.1 - Launching Pctomaty.....	6
2.2 - Anatomy of Pctomaty.....	7
2.2.1 - Main Menu.....	7
2.2.1.a - File menu.....	7
2.2.1.b - Master Menu.....	8
2.2.1.c - Display Menu.....	8
2.2.1.d - About Menu.....	8
2.2.2 - toolbar.....	9
2.3 - Anatomy of the programming page Vimaty.....	10
2.3.1 - Specific area of the current Vimaty.....	10
2.3.2 - Common area for Vimatys of the project.....	11
3 - DEVICES.....	12
3.1 - Create automate device (XMonopro-Multicustom-Minimono-CustoLAN).....	12
3.1.1 - Different types of devices.....	13
3.1.1.a - 0/10Volts.....	13
3.1.1.b - Relays.....	13
3.1.1.c - VCA.....	13
3.1.1.d - Infrared.....	14
3.1.1.e - RS232.....	15
3.1.1.f - MIDI.....	16
3.1.1.g - X10.....	16
3.1.1.h - EIB.....	17
3.2 - Create a Vimaty device	18
3.3 - Actions on devices.....	18
3.3.1 - Actions on 0/10 volts and VCA.....	18
3.3.1.a - Send a command.....	18
3.3.1.b - Test state actions.....	18
3.3.2 - Actions on Relays.....	18
3.3.2.a - Send a command.....	18
3.3.2.b - Test state actions.....	19
3.3.3 - Actions on Infrared.....	19
3.3.3.a - Send a command.....	19
3.3.3.b - Test state actions.....	19
3.3.4 - Actions on rs-232 and Midi.....	19
3.3.4.a - Send a command.....	19
3.3.4.b - Test state actions.....	19
3.3.5 - Actions on X10.....	19
3.3.5.a - Send a command.....	19
3.3.5.b - Test state actions.....	19
3.3.6 - Actions on EIB.....	19
3.3.6.a - Send a command.....	19
3.3.6.b - Test state actions.....	19
4 - DECLARE A VIMATY.....	20
4.1 - Different types of Vimaty.....	20
4.2 - Properties of a Vimaty.....	20
5 - COMMON COMPONENTS TO VIMATY OF A PROJECT.....	22
5.1 - Icons.....	22
5.2 - Macros.....	23
5.2.1 - Edit a macro.....	23
5.2.2 - Macros usage.....	24
5.3 - Memories.....	25
5.3.1 - Declaration.....	25
5.3.2 - Actions on memories.....	26
5.3.2.a - Position a memory.....	26

5.3.2.b - Memory test.....	27
5.4 - Input devices.....	28
5.4.1 - The JPI.....	28
5.4.2 - Analog inputs.....	28
5.4.3 - Actions on input devices.....	29
5.5 - The rs-232 and Midi feedback.....	30
5.5.1 - Principle of recognition.....	30
5.5.2 - Structure of the window of edition of rs-232 feedback.....	30
5.6 - Icons lists.....	32
5.7 - Vimaty EIB Devices.....	33
6 - PROGRAMMING THE INTERFACE OF VIMATY.....	34
6.1 - Introduction.....	34
6.2 - Description of the drawing area toolbar.....	34
6.3 - Windows.....	35
6.3.1 - Create a window.....	35
6.3.2 - Window Home.....	37
6.3.3 - Windows macros.....	37
6.3.4 - Windows actions.....	38
6.4 - Buttons.....	39
6.4.1 - General.....	39
6.4.2 - Handling of the buttons.....	39
6.4.2.a - Change of size.....	39
6.4.2.b - Position icons and text.....	40
6.4.2.c - Move-Align-arrange buttons.....	40
6.4.3 - Push button.....	40
6.4.4 - Bargraph.....	40
6.4.5 - Dynamic texts.....	41
6.4.6 - Hardware buttons of Vimaty.....	41
6.4.7 - Buttons macros.....	41
6.4.8 - Actions on buttons.....	42
6.4.9 - Link a button to a device or a memory.....	43
6.4.9.a - link a pushbutton.....	43
6.4.9.b - Link a bargraph.....	44
6.4.9.c - Link a dynamic button (text).....	44
6.4.9.d - Duplicate a button.....	45
6.5 - Advanced Properties of Vimaty.....	46
6.5.1 - Vimaty macros.....	46
6.5.1.a - Initialization of Vimaty.....	46
6.5.1.b - The JPI of Vimaty.....	46
6.5.2 - Vimatys infrareds.....	46
6.5.3 - Vimatys Fonts.....	47
7 - THE SYSTEM COMMANDS OF VIMATY.....	48
7.1 - Display.....	48
7.1.1 - Date-Clock.....	48
7.1.2 - Message.....	48
7.2 - Memory.....	48
7.2.1 - Reset memories.....	48
7.2.2 - Load memories.....	48
7.2.3 - Save memories.....	48
7.3 - Send command.....	48
7.3.1 - Infrared.....	48
7.3.2 - Fixed string rs-485.....	49
7.3.3 - Memory string rs-485.....	49
7.4 - System.....	49
7.4.1 - Else.....	49
7.4.2 - END IF.....	49
7.4.3 - Insert comment.....	49
7.4.4 - Tempo flickering.....	49
7.5 - Waiting.....	49
7.5.1 - Wait (blocking).....	49
7.5.2 - Wait (not-blocking).....	49

7.6 - Vidéo (70s)	49
7.6.1 - Incrustation (70s) defines.....	49
7.6.2 - Select source.....	50
7.7 - Synchronize Vimaty	50
7.8 - Copy Value Button	50
8 - GENERATE A PROJECT FOR VIMATY	51
9 - THE VIMATY EMULATOR	52
10 - THE PADMATY	53
10.1 - Introduction to Padmaty	53
10.2 - Padmaty page anatomy	53
10.3 - Declare a Padmaty	54
10.4 - Padmaty properties	55
10.5 - Configure the keyboard to help programming	55

1 - INTRODUCTION

PcToMaty makes it possible to create applications for :

- The Vimaty's series touch screen
- Padmaty keyboards.
- XMonopro et Multicustom in mode standalone.

This document presents the Pctomaty usage to make programs for Vimaty's color touch screens : Vimaty 35, Vimaty 35s and Vimaty 70s.

After a presentation of the programming environment, we will broach the various steps necessary to the achievement of a project for Vimaty (Declaration of devices connected to the system, graphic creation, creation of macros).

Afterwards a part concerning an advanced programming of the system like the operation of the rs-232 feedback, dynamic displays...

2 - DEVELOPMENT ENVIRONMENT

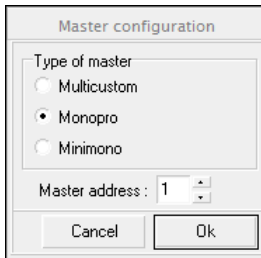
2.1 - Launching Pctomaty

Execute Pctomaty, default located is **Start/Programs/Vity/Pctomaty**.

The welcome window appears, choose **New project...** or **Open a project...**

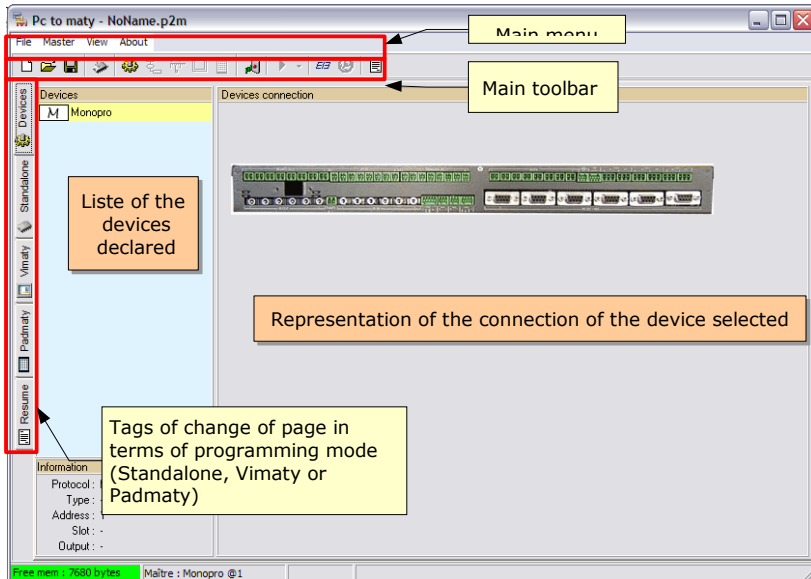


A new window requires the type of master and the master address. This does not concern us. Click on **Ok**.



2.2 - Anatomy of Pctomaty

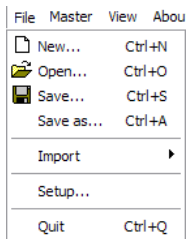
The main window of Pctomaty appears located on the page of creation of devices.



The use of this page is described in the section concerning the declaration of the devices.

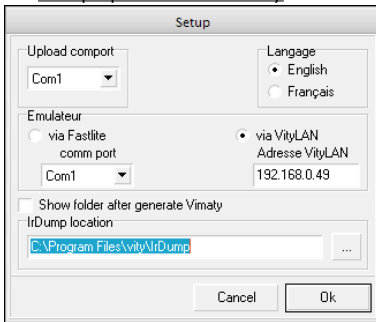
2.2.1 - Main Menu

2.2.1.a - File menu



- **New** : Create a new project.
- **Open** : Load an existing project **.p2m**
- **Save** : Save the project in the current file.
- **Save as** : Save the project with a new name.
- **Import** : Import projects to the current program.
- **Quit** : Quit Pctomaty.
- **Setup**: Configure the software.

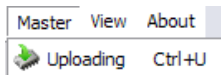
Setup options of Pctomaty



The "setup" window allow you to choose the language, the upload comport **standalone**, the Direct comport of simulation **timeline** and of emulation **Vimaty** and to activate or not the opening of the **VIMATY generation folder**, the location of the management software of the infrared codes **IRDUMP**.

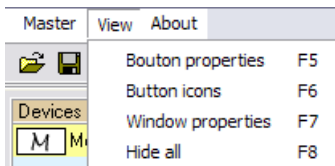
2.2.1.b - Master Menu

This function upload the program into the central unit for a working in mode **standalone**.



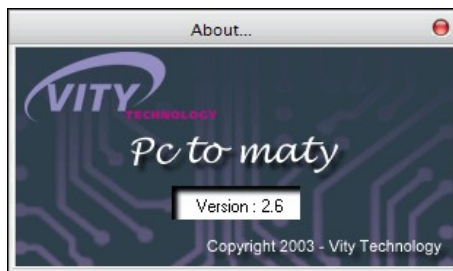
2.2.1.c - Display Menu

This menu display button properties of a window of a program for Vimaty, button selected and down but also the window properties. The last selection makes it possible to hide all opened windows.



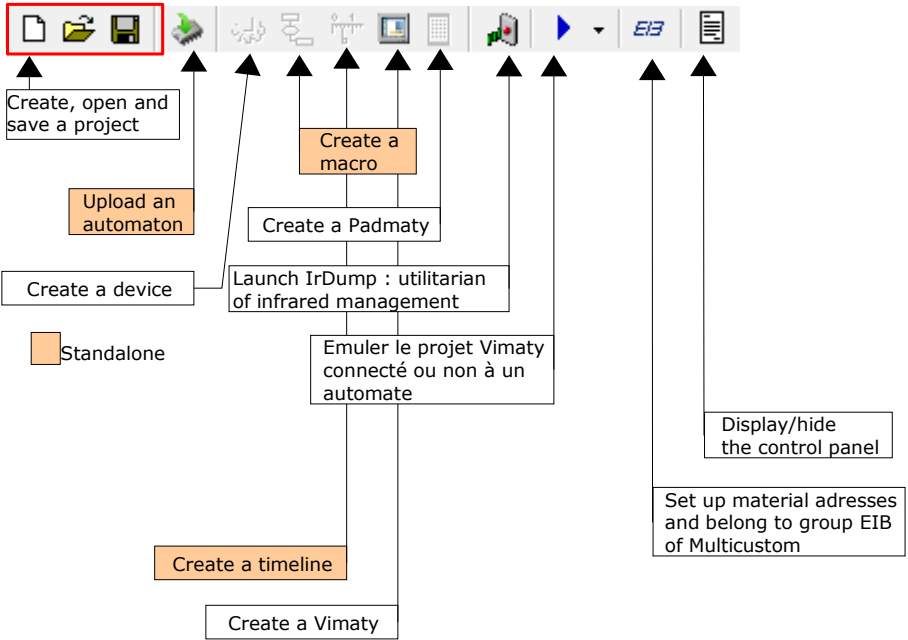
2.2.1.d - About Menu

Open a new window showing the version of PcToMaty installed on your computer PC.



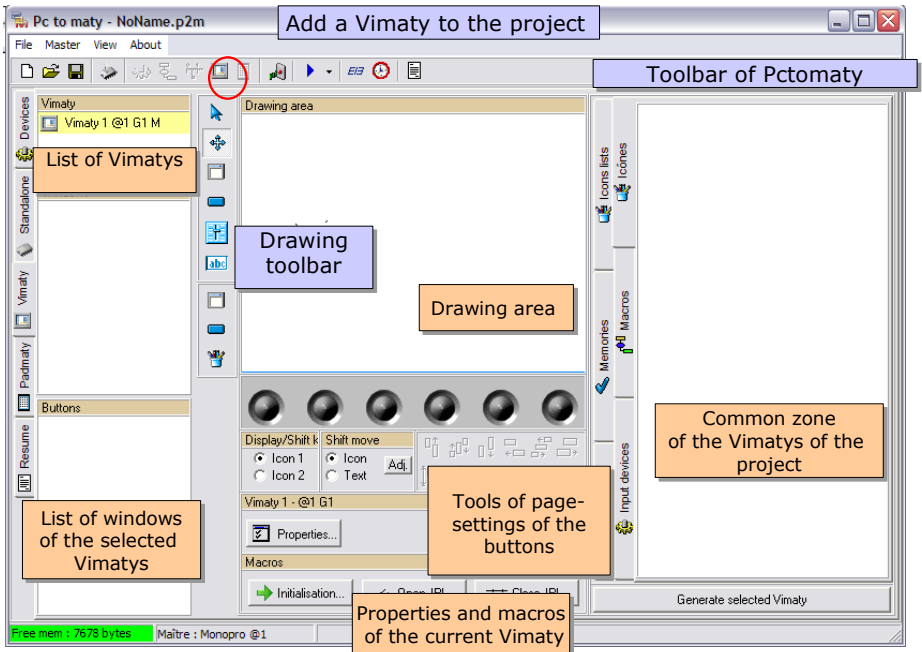
2.2.2 - toolbar

The icons of the toolbar are activated in terms of the programming tag selected.



2.3 - Anatomy of the programming page Vimaty

The Vimaty page is divided in several areas each ones have a precise function. See below a description of these areas.



The programming page of Vimaty includes two great parts: The specific area for Vimaty which is being programmed (on the left) and an area containing the common objects to all Vimatys of the project (on the right).

2.3.1 - Specific area of the current Vimaty

In this part of the window, there is :

- Vimatys list of Vimatys defined in the project.
- Windows list of selected Vimaty.
- The buttons list of selected window.
- The drawing area represent Vimaty screen with drawing tools. Under drawing area, there are press buttons of the Vimaty.
- Under drawing area, management tools of the drawing area (page-setting etc..)
- At the bottom, Vimatys properties, JPI/Infrared management...

2.3.2 - Common area for Vimatys of the project

On the right part of the main window, there are from top to bottom :

- Icons tab (or pictures) used by the Vimaty of the project.
- Macros tab : Macros that can be called by all Vimatys of the project.
- Input devices tab : JPI, analog input or rs-232 feedback are declare here.
- Icons list tab : an icon list it's a set of indexed icons (256 max). Icons lists are use to create dynamic display.
- Memories tab : This tab contains memories declared by the user, the memories state of devices declared in the project : a memory is automatically associated to a device when it's create and the memory contain the device state (ex : 0 for an open relay and 1 for a closed one). Those memories permits to create conditional (test) into macros.
- EIB devices tab : in this tab, EIB devices connected to the Vimaty EIB interface are declare here. These devices are different from EIB Multicustom devices.

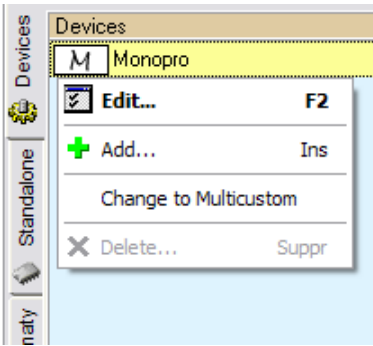
3 - DEVICES

The first step is creation of devices connected to the system. Most of devices controlled by a Vimaty are connected to automate XMonopro, Minimono or Multicustom that transforms rs-485 commands of the Vimaty in infrared sequence, rs-232 data, Relays, Midi etc.. according to the devices declarations. These devices are declared on the page **Devices...**

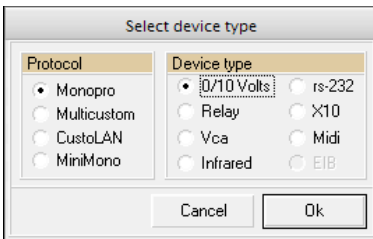
There is a second category of devices : They are directly connected to the Vimaty : Infrared of Vimaty, for Vimaty 35s or 70s the device rs-232 of the Vimaty and finally in the case of a Vimaty 35EIB or 70EIB of the devices EIB declared in the common part of the Vimaty.

3.1 - Create automate device (XMonopro-Multicustom-Minimono-CustoLAN)

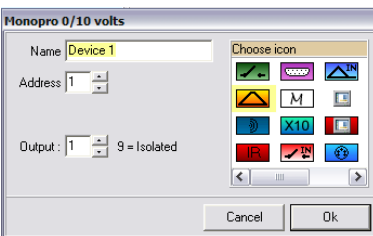
To create a device, select the tag Device,



Right click on the device list and choose **Add** in the contextual menu such as the figure shows.



You must then inform the type of MBC controller connected, the type of devices, and the type of control (relays, infrared, RS 232, etc.). Choose the type among the list and validate.



The definition window appears.

On this window, you must enter the name of your device, the address of your BC controller and the output number of the output on which you have connected your device. You can also change the icon of your device.

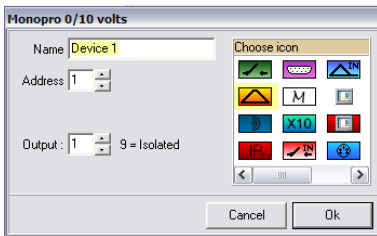
The option macro (for some kind of devices) is only use for standalone programs.



After validation, the created device appears in the devices list. You can edit devices properties again by double clicking on the name or by the contextual menu and the option « Edit ».

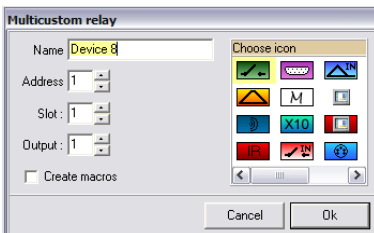
3.1.1 - Different types of devices

3.1.1.a - 0/10Volts



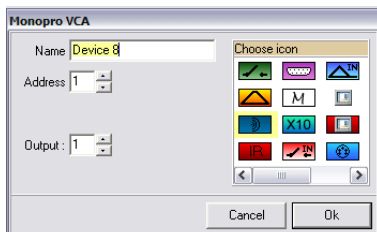
Give a name, an address and an output for your device 0/10volts. If your MBC controller is a XMonopro, The isolated output on 3 pins terminal block instead of two has value **9**. If your MBC controller is a Multicustom, you must precise the slot (position) in which the 0/10 volts module is connected. You can also select another icon for your device in order to quickly locate it in macros.

3.1.1.b - Relays




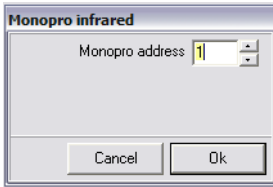
Give a name, an address and the number of the output on your MBC controller for your device relay. If your MBC controller is a Multicustom, you must affect the number of the slot to it. You can also select another icon for your device.

3.1.1.c - VCA

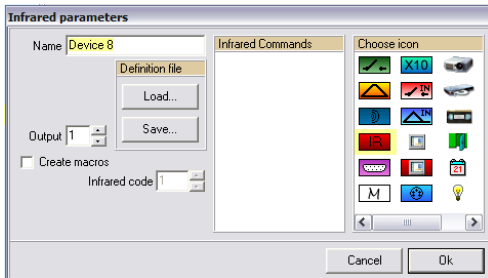


Give a name, an address and the number of your output on which your device VCA is connected. You can also select another icon for your device.

3.1.1.d - Infrared 

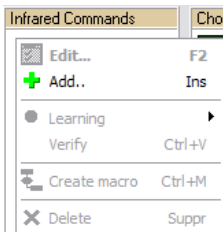


Choose the address of your MBC controller on which is connected the infrared device that you are creating in order to determine or store the codes IR of the device and validate by OK.



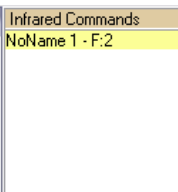
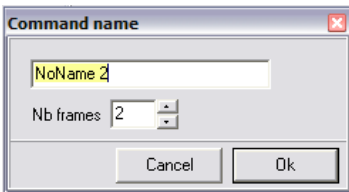
This window appears, you must specify a name and the output number of your infrared device. You can also select another icon for your device. Click on OK to finish.

Infrared codes are store into memory address. You can change this address to match another program (for example made with Fastoch). If the given address is already use, Pctomaty warm you.

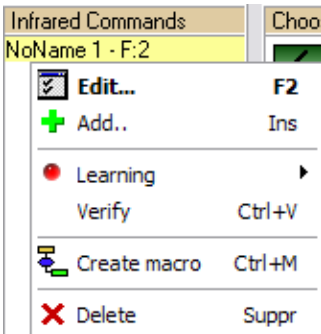


Once device is create, you must add command to declare (ex : play, pause, stop.. for a VCR)

To add an infrared command , right click in the zone «Infrared commands» and select "Add.."



This window appears, you can give a name to your infrared command and define the number of frames that will be send by default.



Now that your command is created you can open the menu **Edit**(F2) while right clicking.

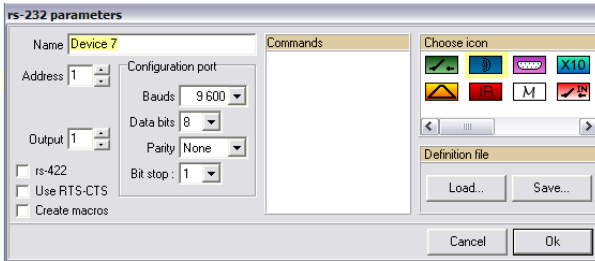
This menu makes it possible to:

- **Edit the name** of your command and to define the number of frames emitted by the infrared output.
- **Learn infrared codes**, in condition of being connected to the power station with the kit of connection RS232/RS485 Fastlite or VityLAN. There are two possibilities of training for the infrared codes: **Normal Mode**: It is the mode to be tested in first which functions for the simple codes. In the event of failure after checking of the code, pass to the following mode:**Forced Mode**: This mode learns the infrared code without analysing them. Generally employed for the training of the complex codes

- **To check the infrared code** previously learned.
- **To create a macro standalone automatically** correspondent with this IR command.
- **Remove the command.**

In the main window, you can also save and load the lists of IR codes under the type of file *.irs. Very useful for the current configurations like VHS,DVD, Satellite etc...

3.1.1.e - RS232 



The window opposite appears. On this one you can define the name of your device, its address and its number of output.

The checkbox rs-422 makes it possible to set up a communication rs-422 on the ports RS 1 and only 2 of XMonopro.

The checkbox RTS-CTS activate the control flow for XMonopro.

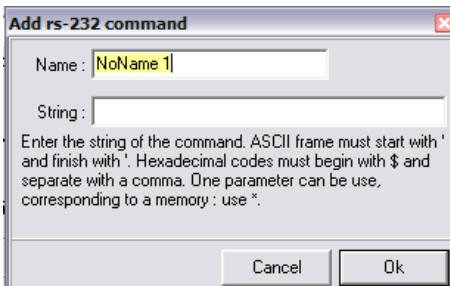
If your MBC controller is a Multicustom you can give the corresponding slot.

In the "Configuration Port" area, it is necessary to define :

- **Rate:** Select the rate of transmission from 300 to 115 200 bauds.
- **Data bits:** Define the number of data bits.
- **Parity:** Define the utilization or not of the parity bit.
- **Bit stop:** Define the number of stop bit.

The window "choose icon" indexes various kinds of icons so as to identify your devices easily.

In the "Definition file" area you can save or load your commands in a file of type *.cfg.



To add a command RS-232, right click in the «command» area and select «Add». In the window opposite, you must define the name of your command. Syntax to be used :

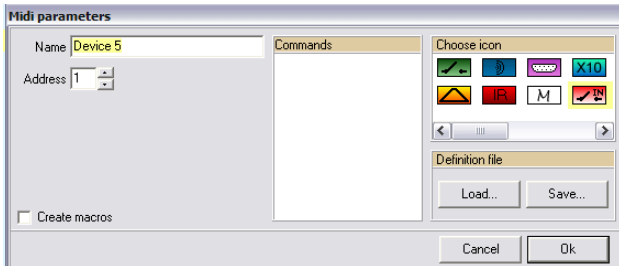
- The parameters in ASCII must be surrounded by apostrophes ('Hello')
- The hexadecimal parameters must be preceded by the sign \$ and finished by a comma.
- You can use a parameter report, to define it use *(star).

EXAMPLES:

- 'PWR ON', \$0d sends the string "PWR ON" followed with a carriage return (\$0d).

- \$01,\$40,*, \$30, 'VP ON' : The parameter report represented by * will be asked to you when you add this command in a macro. The byte * will then be replaced by the value stored in the memory that you will indicate.

3.1.1.f - MIDI

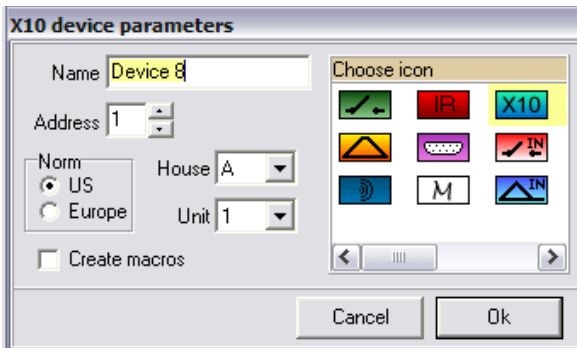


The device MIDI is available only on XMonopro and functions exactly like a rs-232 device.


nb : The administration of rs-232 and Midi feedback are only use by Vimatys and Padmatys and will be treated further.

3.1.1.g - X10

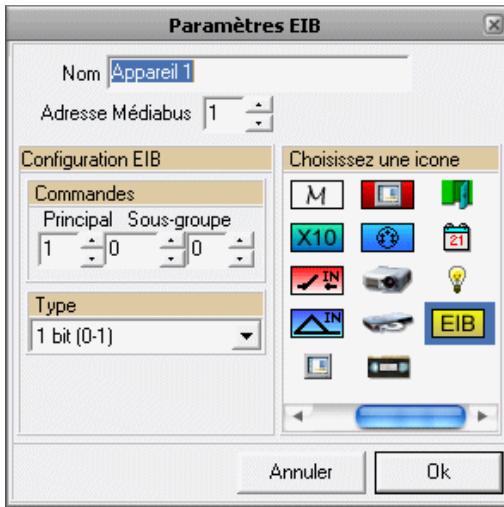
For this type of devices(power line), you must define :



- The name of the device.
- The address of the device.
- The US or European standard
- The home code from A to P.
- The number of the unit from 1 to16.
- The combination Home-unit form the single identifier of X10 device. It is possible to declare several X10 devices
- Even if there is only one x10 output.

3.1.1.h - EIB 

EIB devices are only available with a Multicustom equipped of an EIB module.

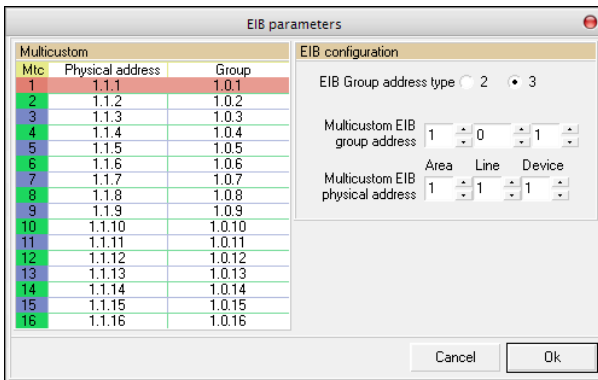


- The name of the device.
- The address of the Multicustom to which EIB module is connected.
- The group address of the EIB device.
- The type EIB of the device.

In addition to EIB devices declaration, it's necessary to give a EIB physical address and a EIB group address to the Multicustom module. For that click on the EIB icon of the main toolbar.

The window which appears allow you to associate an EIB physical and group address to a Multicustom rs-485 address.

The EIB addressing system can work in mode 2 or 3.



You are suppose to know how EIB device work and how they are addressing to use EIB device into Pctomaty/Vimaty/Multicustom.

3.2 - Create a Vimaty device

It is possible to connect devices directly to Vimaty according to their types.

All the Vimatys have internal memory for infrared code (up to 127). Use the infrared button under the drawing area.

Vimaty 35s & 70s can moreover control rs-232 devices. The commands declaration and the configuration of the port are made as same way as XMonopro/Multicustom rs-232. The difference is that the device does not have to be declared : It is automatically reserved according to the type of Vimaty choose. To edit commands, click on rs-232 button under the drawing area.

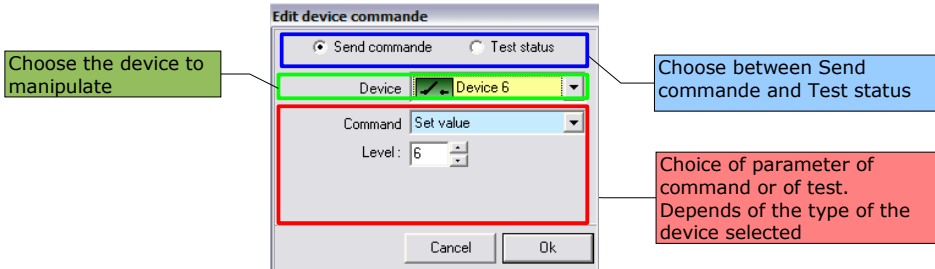
The Vimaty EIB replace a rs-232 interface by an EIB interface and the rs-232 button is replaced by the EIB button which makes it possible to configure the EIB physical address and the EIB group address of the Vimaty. The EIB devices of Vimaty are to be declared in the common area to Vimaty with tab EIB.

The declaration of Vimatys devices will be developed in the section E.IV.

3.3 - Actions on devices

The actions available for a device depend of the type of this one. See in the chapter of the macros for macros creation.

The choice of action on the device is carried out using the dialog box below.



3.3.1 - Actions on 0/10 volts and VCA

The 0/10 volts and the VCA function in the same way.

3.3.1.a - Send a command

The actions on a 0/10 volts or a VCA are :

- **Set value:** position the device on a level given (from 0 to 255). Parameter :The level
- **Ramping:** Gradually position the device with a value. Parameter : The level, the speed of variation.
- **Set with memory value:** position the device on the level of the value of a report Parameter : The memory
- **Increase/decrease** :Increase or decrease the level of the device of the value of a step. Parameter : Value of a step.

3.3.1.b - Test state actions

It is possible to compare the level of a 0/10 Volts or a VCA beside a given value. The possible tests are> (superior), < (inferior) or = (equal). Parameter : A value.

3.3.2 - Actions on Relays

3.3.2.a - Send a command

The relays can receive 3 commands: Open, Close or Click. Click close the relay during ½ second then open it. There is no parameter.

3.3.2.b - Test state actions

It is possible to test if a relay is closed or open..

3.3.3 - Actions on Infrared

3.3.3.a - Send a command

The commands being able to send by an infrared device are those definite at the time of the declaration of the device.

The parameter is the number of frame to send from 1 to 254. A number of frame equals to 0 stop sending of the code, and value 255 to emit the code uninterrupted.

3.3.3.b - Test state actions

The test of state of infrared are made by comparison of the state of the device with one of its command. The test of equality makes it possible to know if the device emitted a command, the tests of superiority and inferiority determines if the device has sent a command declared before or after the current state in the list of its commands.

3.3.4 - Actions on rs-232 and Midi

The devices rs-232 and Midi function in the same way.

3.3.4.a - Send a command

The commands being able to send by a rs-232 device are those define at the time of the declaration of the device.

If the character * is present in the string to be sent, it will be replaced by the value of the memory given in parameter.

3.3.4.b - Test state actions

The test state function works like infrared device.

3.3.5 - Actions on X10

3.3.5.a - Send a command

The X10 can receive 4 commands: On, Off, Dim or Bright (close, open, diminish and augment). There is no parameter.

The reactions depend on the type of X10 device recipient.

3.3.5.b - Test state actions

It is possible to test if a X10 is on or off.

3.3.6 - Actions on EIB

The actions on EIB depend on the type of declared EIB.

3.3.6.a - Send a command

For a EIB 1 bit: The commands are Stop/Open/0 and Running/Close/1

For a EIB 4 bit : The commands are positioning of level compared to the current position.

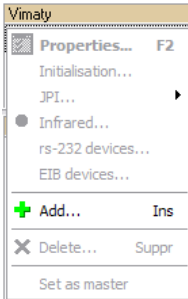
For a EIB 8 bit: The commands are to bring to the level of a memory or To level given.

3.3.6.b - Test state actions

The test of a EIB 1 bit functions like a relay, the test of the 4 bits and 8 bits are made compared to a level like one 0/10 volts.

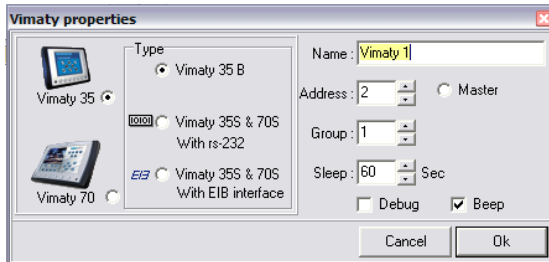
4 - DECLARE A VIMATY

For any handling of the elements specific to Vimaty, it is necessary to create at least Vimaty; for that, reach the page of programming of Vimaty, select the VIMATY tab.



To create a touch screen Vimaty, right click on window VIMATY located on the left of the main window and choose Add.

To declare Vimaty, click on the eighth icon of the Vimaty toolbar or call the contextual menu of the list of Vimaty and choose the Add option... The Vimaty properties window appears:



4.1 - Different types of Vimaty

- The Vimaty 35B : connection rs-485 for piloting of Monopro/Multicustom.
- The Vimaty 35s & 70s : Same characteristic as 35B with an interface rs-232 integrated in version screen LCD Color 3"5 and 7".
- The Vimaty 35EIB & 70EIB : Same characteristic as 35B with an interface EIB integrated in version screen LCD Color 3"5 and 7".
- All Vimaty have in common an infrared interface, an input JPI, an intern infrared sensor and external.

4.2 - Properties of a Vimaty

- Its name.
- Its address that must be unique.
- Master : In a project, only one Vimaty can be the Master but one needs at least one of them.
- Its group.
- The time of deactivation (time before the cut of the retro-lighting of the screen).
- Operation in debug mode or not (posting of messages in the event of error).
- Quiet screen or not.



In Vimatys list, on the top-left of the main window, the name of the Vimatys, address, group are shown. The letter M defines the Master screen.

The concept of group makes it possible "to group" Vimaty together. When of Vimaty "are grouped", it is possible to send a group command which will make react all Vimaty of the group specify by the command in the same way (for example all Vimatys of group 2 posts window 1).

Take care : Vimatys and Padmatys division the same space of addressing. Contrary to XMonopro and Multicustom which can have the same rs-485 address, Vimaty and Padmaty cannot do it.

That results in two

- An address rs-485 used by Vimaty cannot be used by Padmaty and conversely.
- There can be one Master : If a Vimaty is a Master, no Padmaty can be it and conversely.

5 - COMMON COMPONENTS TO VIMATY OF A PROJECT

The common elements to Vimaty are laid out in the tab on the right the page of Vimatys programming. They are usable by the whole Vimatys of the project.

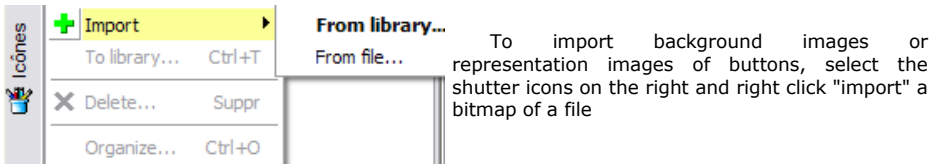
Vimaty manage, in addition to the devices of automate seen previously, icons (or image) with format BMP Windows, macros (group of command on various objects), information feedback (JPI, analogical input, return rs-232/Midi), memories defined by the user and devices directly connected to Vimaty (rs-232, infra-red, EIB). Vimaty can control devices without additional control interface for small project.

The following chapters describe the role of each tab of common part for Vimaty of the project.

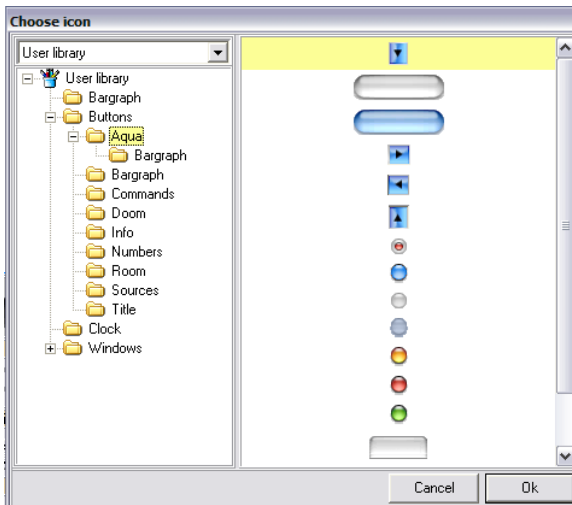
5.1 - Icons

The programming of Vimaty requires the use of images (icons). Icons are used to draw buttons of an application or to customize of windows...

To use an icon it must be first imported from a file or from the Pctomaty icons library.



This option opens access to the Windows explorer to select the image you want to import.



You can also use the icon libraries published by VITY Technology which contains a great choice of bitmaps and background images. For that, after downloading the library on the VITY Web site, double-click in the list of the icons.

Open combobox library to access to new libraries such as Aqua, Gold, Metal...

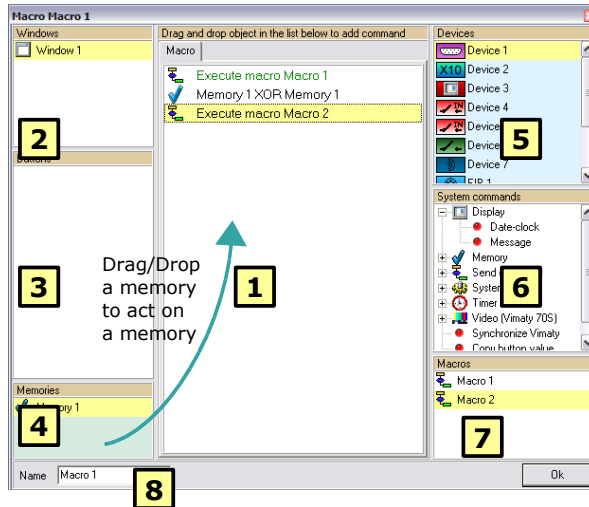
Imported icons can be used for buttons or windows by "drag&drop" icons into the window/button area.

5.2 - Macros

Vimatys macros allow you to create macros command which can be used by all Vimatys/Padmatys of the project.

5.2.1 - Edit a macro

Select VIMATY tab on the left of the screen, then Macros on the right of the screen. Call the macro contextual menu and select "add...". The macro window is displayed with around lists of available objects for Vimatys programmings. Windows, Buttons, Memories, Devices, Commands Systems, and also macros already created. Drag and drop the objects in the zone in the center to create your macro. Don't forget to give a name to your macro in bottom of the window **8**.



To add action into a macro, drag-drop from the object on which one wants to act (window, devices...) towards the central area **1**.

To edit a command, double-click on this command or call its contextual menu. Via the menu it is possible copy, paste, delete, edit... actions.

Description of the window of edition of the macros

1 This list contains actions of macro in editing progress, it is possible to reorganize the actions by Drag'n'Drop to move, to copy-cut-paste action of macro to another macro or into the same macro. The rules of multiple selection of the command lines works like the standard multiple select lists in Windows (the operating system).

2 The Windows declared in Vimaty in progress makes it possible to add action on windows which thus make it possible to change pages.

3 The buttons list contained in the window currently opened in the area of drawings makes it possible to add action on the buttons such as for example changing its icon.

4 The memories user list makes it possible to act on the memories declared in the project. It is thus possible to change the memories value, to copy a memory into another, to test value, to copy the state of a device in a memory, etc...

5 The devices list of the project : it is by a drag'n'drop that one can send an order to a device. For example to send the Play command to Vcr. It is also possible to test the state of devices.

6 The systems commands list : in this list, commands known as system, such as the display of a video window (Vimaty 70s and EIB), Remove all the memories, the timer (times). The whole of these commands will be described further.

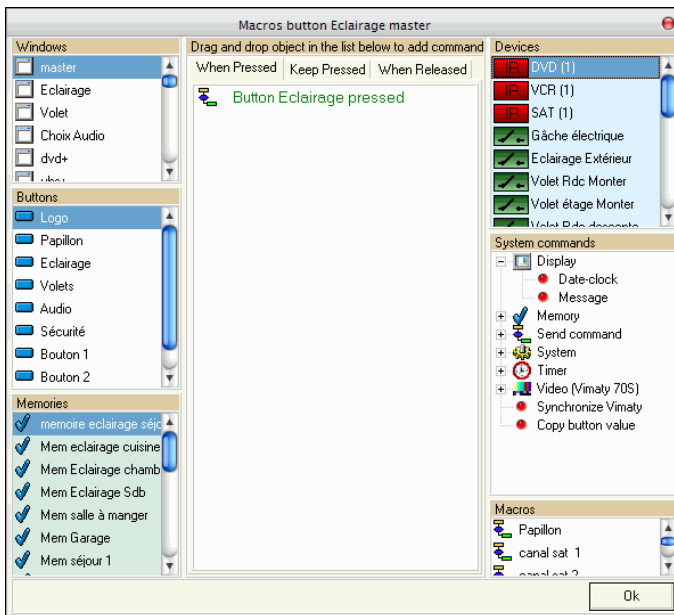
7 The macros list : to call a generic macro, drag'n'drop it into the central area.

The window of edition of the macros can have several central tabs and contain up to 3 macros. Several tabs are used to edit the 3 macros of buttons : there is a macro call when the button is pressed, a macro is call when the button is keep pressed and a macro is call when the button is released release. There is also 3 tabs for windows macro editing. All this will be describe in the graphic part of the development.

5.2.2 - Macros usage

The macros defined in the common area of Vimaty, is intended to be called by another macro.

To insert the call to macro, drag'n'drop a macro from the list of the macros **7** towards the area **1**. It is possible to reedit the line add to the actions by double-clicking above a selector appears making it possible to change the macro in the event of error.



Example of editing macro window with button tabs on top

5.3 - Memories

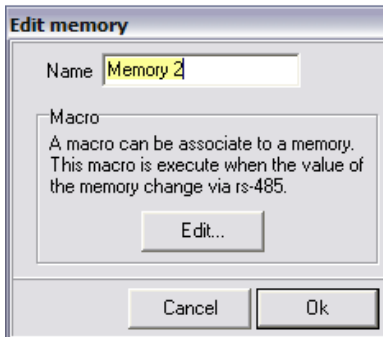
5.3.1 - Declaration

Vimatys memories make it possible to store values and create conditional programming.

Ex: if memory 1 is worth 2
 then close relay 1
 else open relay 2.

All the memories can take any value between 0 and 255 included, either 256 different levels.

To reserve a memory, select Vimaty tab on the left screen, then Memories on the right of the screen. Call the contextual menu of the memories list and select "**add...**".



The memory window is displayed. You can give the name to the memory.

Click on the button **Edit** to open the macro editing window. This macro will be call automatically when the memory value change. This is ideal to generate actions related to a change of value of this memory. The Macro logo is displayed near the macro name in the list when the macro of the memory have actions.

Each VIMATY has **240** memories.

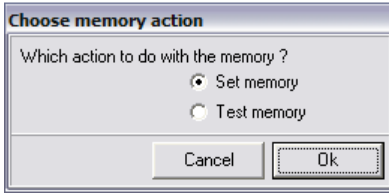
Each device uses a memory to store its state, the remainder is available for the user. This is why in the list of the memories the devices and Vimatys appear because these objects use a memory. You can also use a macro for each change of device state. The macro logo is registered next to it.

In the window of declaration of the memories, there is a "Macro" button, this button makes it possible to edit macro which will be carried out with each time the memory changes value.

The memories associated with a device have the same capabilities, which makes it possible to start macro automatically when the state of a device changes.

The principal interest of this functionality is the simplicity of programming external events with the system : for example, if the fire alarm starts then the light are switch on, projection stop and a message is displayed on the Vimaty screen, to do that, connect alarm to a JPI and add commands into the macro associated to JPI memory the memory (Without forgetting to test if alarm is extincted or ignited).

5.3.2 - Actions on memories

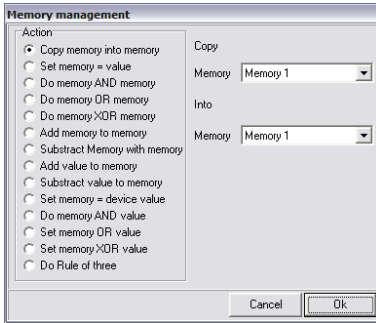


The memories have a significant role in the macros, they make it possible to set up conditional programming. One can either change the contents of a report (to position the memory) or test the value of a memory.

When one adds an action on memory in a macro, Pctomaty requires to choose between Position the memory or test the memory.

5.3.2.a - Position a memory

The first of the possible actions on a memory is to give it a value or to position a memory. After validation of the action window, the window form memory setting is displayed as follows:



The possibilities of handling memory is wide, in all the cases the memory destination contains the result of handling.

The parameters of a handling are either another memory, or a numerical value, or the state of a device.

The Handling or operation to be carried out is either a recopy, or an arithmetic operation or or a binary operation.

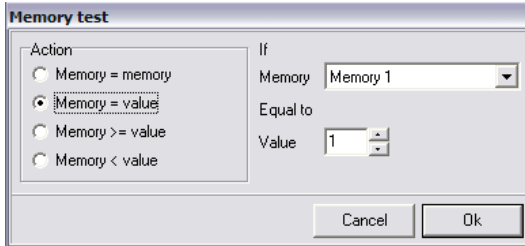
Description of the actions of positioning memory

- **Copy memory into memory** copy the value of the memory source in the memory destination : the two memories are to be specified.
- **Set memory= Value** put a value given in a memory. The memory destination and the value are to be specified.
- **Do memory AND memory, Memory OR memory** and **Memory XOR memory** calculate respectively the binary operation AND, OR or XOR between the memory source and the memory destination with the result in the memory destination. The two memories are to be specified.
- **Add memory to memory** and **Substract Memory with memory** calculate respectively the sum or the difference between the memory destination and the memory source with result in the memory destination. The direction of the operation is destination – source.
- **Add value to memory** and **Substract value to memory** identical to the preceding action but the memory source is replaced by a value to specify.
- **Set memory=device value** copy the value of a device in a memory.
- **Set memory AND value, Set memory Or value** and **Set memory XOR value** carry out a binary operation between the memory destination and a value to be specified.
- **Do rule of three** calculate a rule of three applied to the memory source with result in the memory destination. The parameters multiplier and divider are to be specified. Easy way: By positioning the multiplier with 1 there is a division operation and by positioning the divider with 1 there is an operation of multiplication.

In all the cases the values sources and destination are entirely ranging between 0 and 256, the higher overflow give the result 255 and lower overflows result 0.

5.3.2.b - *Memory test*

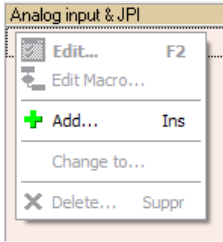
The tests of memory use a memory and a value or two memories.



Description of the actions of memory test

- **Memory=memory** test equality between two memories.
- **Memory=value** test the equality enters a memory and a value.
- **Memory>=value** corresponds to inferior or equal and **Memory < value** to inferior and this between the memory and a value.

5.4 - Input devices



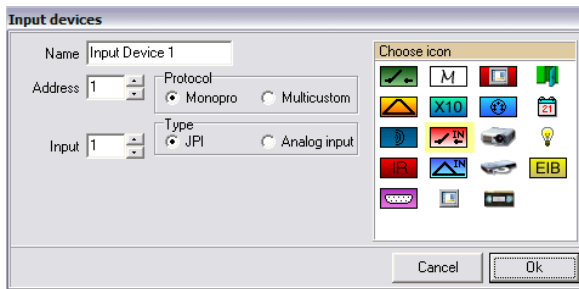
The devices known as "of input" make it possible Vimaty to react to events outside to the system. It gathers two new types of device : the JPI (dry contact) and Analog Input (analogical input). The only actions possible with these devices are the tests of states or the recopy of their value in a memory.

There are on this tab the rs-232 and Midi feedback which make it possible to act according to a received frame.

To create an device entering applications VIMATY, click on the tab "input devices" on the right of the screen. Click on the right in the area "analog input and JPI"

5.4.1 - The JPI

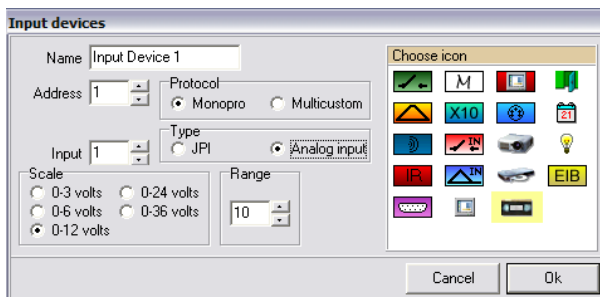
By default the device type is a JPI (dry contact) of which you must position the type of device, the address RS485 and the number of entry.



These devices appear in the list of the memories because they use memories to store their states. It is possible to bind to macro which will be carried out when they changes state.

5.4.2 - Analog inputs

The box Analog Input creates machine of analog input on which you must select the gauge. The value of the threshold by default is 10 on scale from 1 to 255. This is the minimum value the input must change (+ or -) to generate a message. The goal is to limit the number of messages on MediaBus.



These devices appear in the memories list because they use memories to store their states. It possible to link a macro which will be call when the memory change.

5.4.3 - Actions on input devices

The input devices can only be tested. When an action is added on a JPI or an analog input, the following choice is proposed:

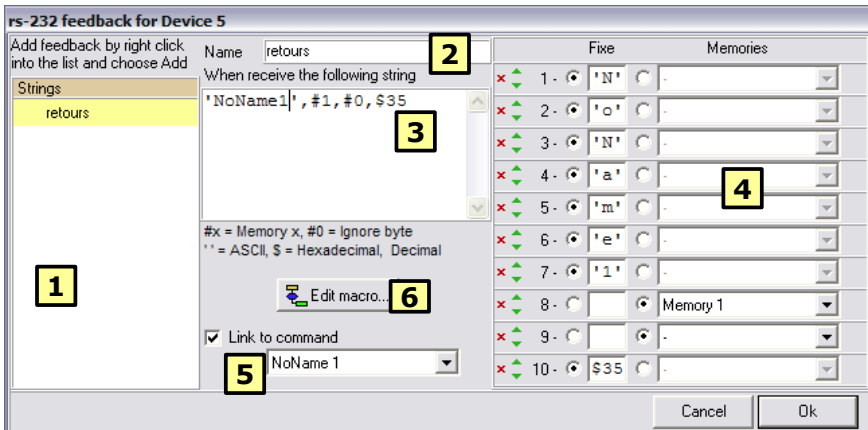


In the case of the AnalogIn devices, it is possible to choose between the test of superiority, inferiority or equality with a value.

For the JPI the choice is limited to Open or Close. The test will be true if the JPI state is equal the selected state.

5.5 - The rs-232 and Midi feedback

If devices rs-232 or Midi are declared, they appear in the lists of rs-232 feedback (below input devices). To manage the rs-232 feedback of a device, call the contextual menu and choose the single option **Edit feedback...**



This window makes it possible to describe the format of the strings to be supervised as well as the behavior the Vimaty must have when a string is recognized.

5.5.1 - Principle of recognition

A string "is recognized" as being a declared feedback if it match the following conditions:

The string received have the same length as the format : maximum 15 byte for Multicustom and 30 for XMonopro.

The fixes characters (bytes) are identical to the format.

The ignored bytes and the bytes memorized with the sites are exactly described in the format.

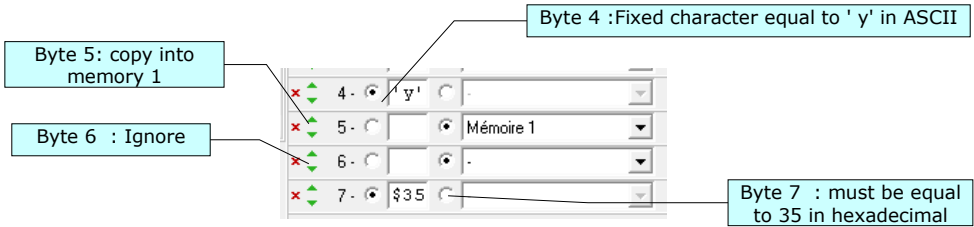
5.5.2 - Structure of the window of edition of rs-232 feedback

Part **1** is the list of the string to scan. To add a string, call the contextual menu of the list then the option "Add", that creates a new string which has as a name **NoName**.

Give a name **2** then define the format of the string in **3** and **4**.

The part **3** makes it possible to declare the string as for the declaration of a command rs-232 with two additional possibilities : be unaware of a received character (represented by #0) and copy the value of a byte received in a memory (represented by #x where x is the index of a memory user).

The part **4** is a complement of the preceding part in order to contribute to the construction of the string to recognize (description following page).



To choose between the fixed mode or memory/ignore, it is needed to select the corresponding radio-button.

The red crosses is to remove the byte concerned of the string, the green arrows respectively make it possible to add a byte to be recognized before or after the byte concerned.

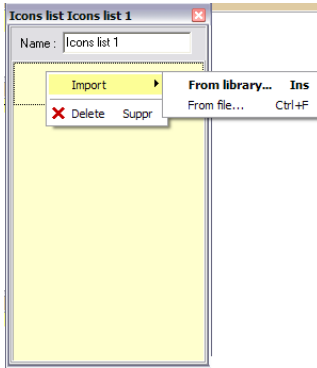
The parts **3** and **4** fill at the same time. If bytes present in the part **3** do not appear in the part **4**, there is a syntax error in the text area and the incorrect bytes will be ignored.

The check box **5** makes it possible to link a feedback to a device command. That makes automatically change the device state according to the received string. In the example above, when 'Play',x,x, \$35 is received (where x are unspecified bytes) the device is regarded as being in the state "Play"

The edition of the macro **6** makes it possible to edit a macro. This macro will be execute automatically when the associated string is recognize. The macro icon appears on the left name of the string in the list if the macro associated contains commands.

5.6 - Icons lists

The Icons lists tab is use to create lists of bitmap which will be used to animate a button or changing the display to a memory value or device state.



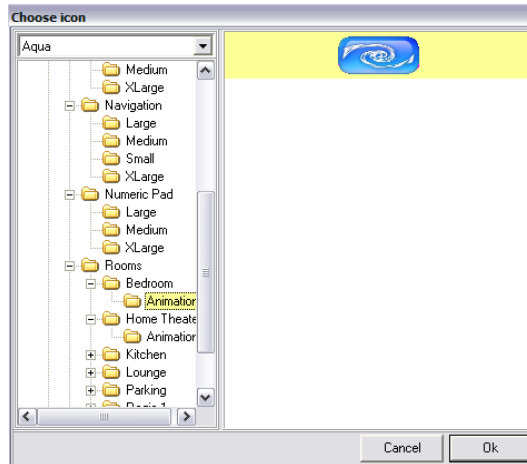
Select the tab "**list icons**", call the contextual menu to create a new list.

It is possible to add images in an icons list by loading a file, the library of icons or by drag-drop images starting from the icons list tab.

Each icons list can receive 256 indexed images from 0 to 255 (possible values of a memory or a device).

Ex : if the memory or the device (for example a 0/10 volts) with value 1, the button linked to the device, in the button properties window, will display the icon 1 of the associated list.

You can load the lists of icon for the animation of the buttons by calling upon the library edited by VITY, like above.



5.7 - Vimaty EIB Devices

The commands intended for devices EIB declared in the common list of Vimaty will be sent via interface EIB of Vimaty or Padmaty. It is necessary there is at least one Vimaty or Padmaty EIB to be able to use the devices declared here.

The declaration of EIB devices for Vimaty/Padmaty is identical of Multicustom EIB devices .

These devices appear in the list of the memories because they use memories to store their states. It is thus possible to bind to macro which will be carried out when they changes state.

6 - PROGRAMMING THE INTERFACE OF VIMATY

6.1 - Introduction

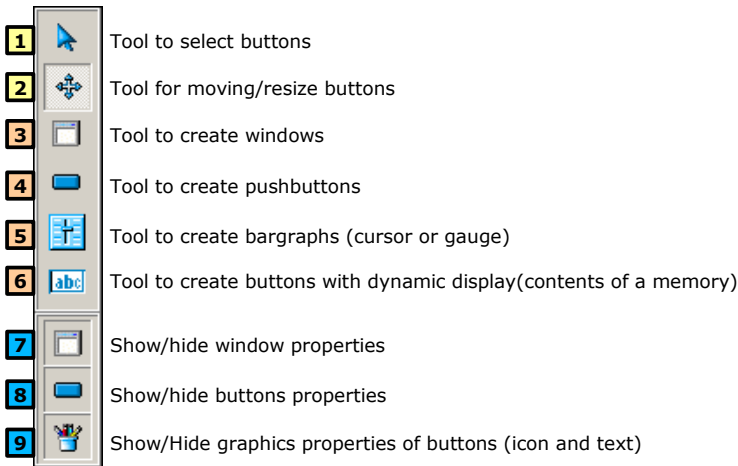
To make application for Vimatys, you must create the user interface with graphics. Only limit graphic are the size of the screen (obviously) and the capacity of CompactFlash which will contain the project.

The user interface is made up of graphic windows (or pages) with or without picture background, a window contains buttons : pushbuttons, bargraph and pushbutton with dynamic text (value of a memory or a device).

In addition to the graphic buttons, you can configure each physical button (hardware buttons) at the bottom off the screen area of the Vimaty.

One color is reserved for transparency : fuchsia (RGB : 255-0-255). Any pixel of the image having this color will not be displayed, thus leaving the visible background.

6.2 - Description of the drawing area toolbar




Tools **1** and **2** manage buttons into a window. The first by selecting buttons and the second by moving them and/or Re-size. You will find a description complete of these tools in the buttons chapter.

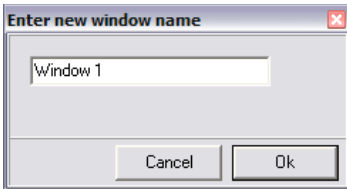
Tools **3**, **4**, **5** and **6** respectively make it possible to create a window, a pushbutton, a cursor and a dynamic button. To create a button, select the desired tool then trace a framework in the zone of drawing.

Tools **7**, **8** and **9** show or hide respectively the windows properties, the functional properties of button and graphics properties of buttons. These window can be displayed permanently during the creation of the project, their content is automatically updated according to the selected object.

6.3 - Windows

6.3.1 - Create a window

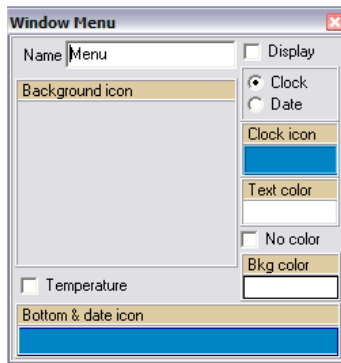
To create a window, click on the tool "create window" .



A news dialog box is displayed to specify the window name.

A window VIMATY is always the same size of the screen.

Validate, the window properties window appears, allowing you to change the appearance and the name of the selected window.



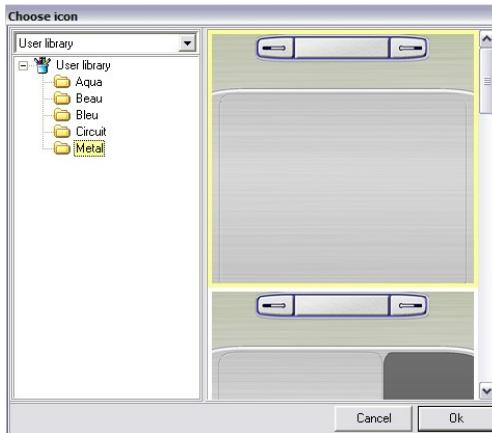
Window properties

The **background icon** area is for a the background image of the window. To insert a background image call contextual menu (right click) of the background icon area. In the menu which appears, you have two possibilities :



load an image in bitmap type (* bmp) presents on your computer

load bitmap for available library installed on your computer.



Example of choice by a library.

It is also possible to drag'n'drop an image from the icons list to the Image area of the properties window.



If the background image is smaller than the size of the window, you can move it with the mouse in the drawing area, after having to activate the displacement mode in the toolbar:



You can also make profitable the color fuchsia transparency (RGB=255,0,255) in the bitmap. In this manner, in the place of the pink area, it is the bottom of another window (nontactile) which will appear.

The background image can be removed to have a window plain of the color defined in the box **background color**. If the color is the color of transparency, the window will not have background. If the box **Empty** is selected, the areas of the window which do not contain an image will appear in "grayed" of the prime coat.

Text color makes it possible to change the color the date, the time and the temperature.

To display the date and/or the time, select the box **Display** then choose between **Clock** and **Date**.

Temperature can be displayed by selecting the corresponding box.

Each one of displays has its own background image. The choosing an image is done in the corresponding area in the same way as for the background image of the window.

- The clock use the image defined in **Clock icon** (size by default 70x22.pixels)
- **Date and clock** use the image **bottom & date icon** (size by default 320 x 22 pixels)
- **The Température** measured by the internal sensor with the VIMATY (same background than the clock).



Display clock and temperature



Display date

Note : The window properties is for current editing window.

Copy windows

To quickly create a copy of a window, drag'n'drop from the window list to drawing area.

If the window you want to copy is in another Vimaty of the project, do the following :

- Select the Vimaty containing the source window
- Select the window you want to copy into the window list
- Call the contextual menu and select "**Copy window**"
- Select the Vimaty you want to copy into
- Call the window list contextual menu and select "**Paste window**"
- Pctomaty requires a name of window
- The window is copied with all its buttons and its properties.

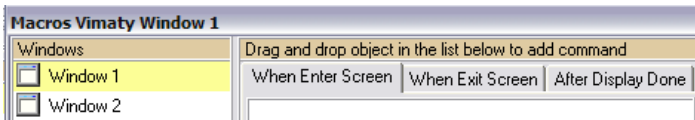
6.3.2 - Window Home

One window of a project is the home window. It is the window which will be displayed first when application start into the Vimaty.

To define a window as Home, open the contextual menu of a window and select the option **Set home window**. The name of the selected window is then followed with (home).

6.3.3 - Windows macros

Windows can call macros when changing state. Select the option **Edit macros** in the menu of a window. Edition box of macro seems below.

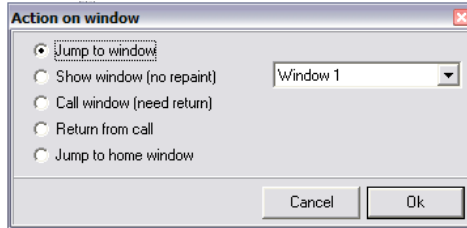


Three tabs in the macros editing window show the 3 macros of a window:

- **When Enter Screen** : This macro is call when the window is displayed but before it's visible.
- **When Exit screen** : This macro is call when another window is displayed.
- **After display done** : This macro is call after buttons of the window are visible.

6.3.4 - Windows actions

Macro can act on windows in the several way. Do a drag'n'drop from a window towards the central area of the macro editing window and the following window appears:



The actions are:

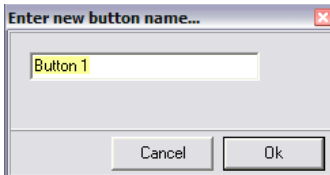
- **Jump to window** : the window is displayed such as define.
- **Show window (no repaint)** : the window is displayed without the background icon. This function makes it possible to display a window without "flickering" when changing page.
- **Call window(need return)** : The window is stack on the current window. That makes it possible to go back to the previous window by calling the function **Return from call**. It is possible to stack up to 16 windows.
- **Return from call** : Complementary of **call window** function.
- **Jump to home window** : Go back to home window.

6.4 - Buttons

6.4.1 - General

To create a button in a window, click on one of these tools **4**, **5** or **6** in the toolbar of the drawing area.

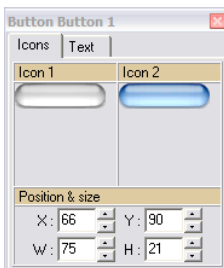
Draw the button shape in the drawing area (the cursor represents a cross then), for that, position the cursor of the mouse in the drawing area, press and hold the mouse left button, move the mouse. When the shape of button is drawn, release the mouse button.



Pctomaty ask for a name to your buttons, then click **OK**.

Buttons properties windows appear. It is possible to modify the graphics parameters for buttons.

Button properties windows can stay visible or not. To show or hide these windows click the toolbar button or press F5/F6 on keyboard.

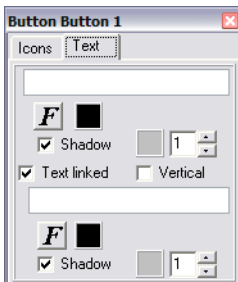


Button icons

Buttons have two icons (icon 1 represents the button in a normal state (released) and icon 2 represents the button in an active state (pressed)).

Display icon 1 or icon 2 depends on the actions execute by macro. Detailed description of the behavior of the buttons will be seen further.

Information under the icon of this window gives the position of the button in the window and its size. It is possible to adjust these parameters here.



Text of Button

Text of button works in the same way as icons : text 1 define the text and the shape button in a released state and text 2 in pressed or activated state.

If texts are identical, select the box "Texts Linked" which will automatically copy the field from top to bottom. You can separately modify the police, the color of the text, the presence, the color and the thickness of the shade. You can also display the text into vertical.

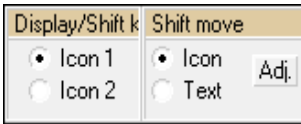
6.4.2 - Handling of the buttons

6.4.2.a - Change of size

Choose the tool **2** in the toolbar and click on the desired button.

A frame rectangle is displayed around the button with handles. Place the mouse cursor on one of these handles : it changes. Click on the mouse left button and move the mouse holding the button. The frame of button follows the mouse. This framework represents the tactile area of the button, it can be smaller or larger than the icon. It is also possible to change the size of the tactile area with the arrows of the keyboard combined with the shift key.

6.4.2.b - Position icons and text



Under drawing area there is a selection box making possible to choose on what the cursor of the mouse will act.

The choice between icon 1 and icon 2 makes it possible to visualize icons 2 and text 2 of the buttons and to handle them.

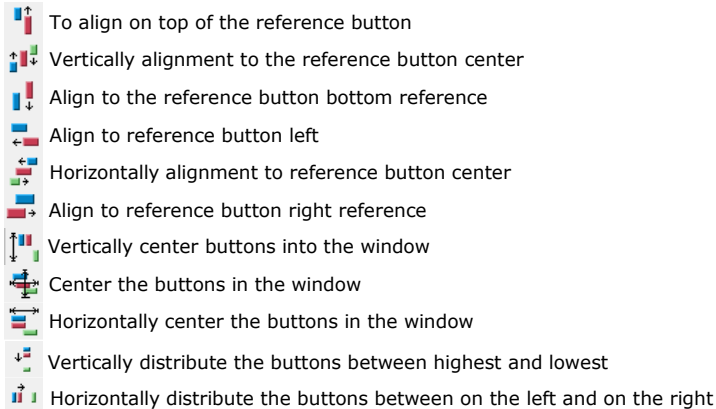
To move the bitmap inside the button, select **icon** in the **"Shift move"** area in bottom of the drawing area, then hold the SHIFT key pressed. The link between the center of the button and the icon is shown with a green line.

To move the text in the button, do in the same way with the option **"Text"**. A red line between the center of the button and the center of the text is shown with a red line. To bring back the button to the size of the icon and to center the text, click on button **ADJ.**

6.4.2.c - Move-Align-arrange buttons

To move and/or align a series of buttons start by selecting them by maintaining pressed CTRL key or by using the tool for selection **1**. Selected buttons appear with a grid. The group of buttons created can be moved with mouse or with keyboard.

For alignment, the reference button is the button which has handles. Buttons with grid will be aligned to this one



6.4.3 - Push button

Push buttons are buttons with two states which change aspect (display of icon 2 and text 2) according to events such as an command in macro, device change state or a memory.

6.4.4 - Bargraph

The bargraph or cursor are buttons with background icon and another icon representing the cursor. The position of the cursors reflects either the state of a device or the value of a memory. This type of button makes it possible to handle easily of the 0/10 volts.

Draw it on the working area, on the window which appears, specify the name of the cursor.



Icon 2 of a bargraph is the cursor. The position of icon 2 represents value 0. The maximum position of the cursor is given by the properties of the bargraph when it is associated to a memory or a device.

The bargraph have particular representation **gauge**. In the case of a gauge, the 2 icons have same size exactly and Vimaty displays a number proportional of lines or column (according to the orientation of the bargraph) to the value which the button represents.

6.4.5 - Dynamic texts

The dynamics texts are normal buttons with the possibility to display a value contained in memories or device value. They are called dynamic because their contents change automatically according to the value of the object which they represent.

Icon 2 of this type of button does not exist, on the other hand they have 2 types of text fixed like the other buttons and a dynamics which represents the value followed by the button. The dynamic text can be repositioned as if it were a second icon.

The precise use of this type of button will be seen further.

6.4.6 - Hardware buttons of Vimaty

On the touch screen, Vimaty has 6 hardware buttons. These buttons can be programmed like buttons of the tactile area. Macros of these buttons is link to a window, there behavior change according to the displayed window.

6.4.7 - Buttons macros

Buttons have 3 macros:

- **When pressed** : call when one presses on the button
- **Keep Pressed** : call while the button is pressed
- **When released** : call when the button is released.

Drag and drop object in the list below to add command




When Pressed | Keep Pressed | When Released

When pressed, keep pressed and **when released** : select the tab you want to add actions. Actions can be moved from one tab to another by drag'n'drop.

Repeat have two parameters in bottom of the window:

Start delay 50 cs Repeat delay 75 cs

The first introduces a time before execution of the actions contained into the tab keep pressed. The default value is 0,5 second. The second value indicates the time between each execution of the macro. The default value is 0,75 seconds.

When a button defined actions in one or more of its macros, the symbols ,  and  area show at the left bottom corner of the touch area of th button.

 : Indicate that the macro **When Pressed** contains actions.

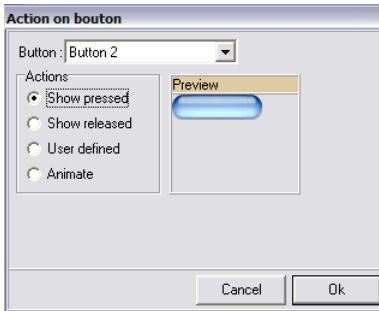
 : Indicate that the macro **Keep Pressed** contains actions.

 : Indicate that the macro **When released** contains actions.

The same symbols appear in top on the left of the physical buttons of Vimaty.

6.4.8 - Actions on buttons

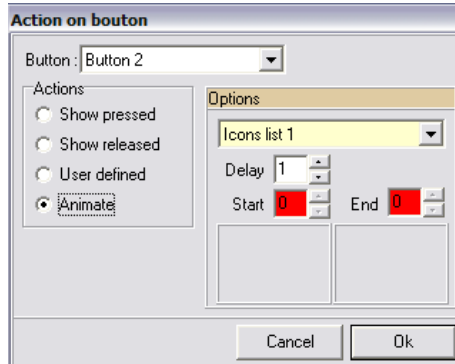
It is possible to do 4 type of action on a button:



- **Show pressed** : display icon 2
- **Show released** : display icon 1
- **User defined**: Pctomaty requires which appearance must take the buttons while proposing to choose in the list of the icons of the project.
- **Animate** : Make possible to do an animation of the button using an icon list.

Animate a button

To use an animation of button it is necessary as a preliminary to build a list of images or to load some from the library.



Select the icon list to be displayed, the delay between 2 images of animation, index of start image and end image index.

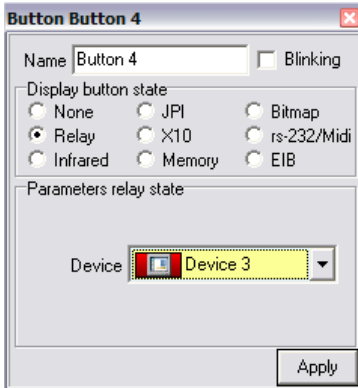
Button will display successively the images between the beginning to the end. If the index appear in red, that's means the start image is the same as the end image and that should be corrected.

The beginning can be of an index higher than the end to make the animation play in the back direction.

6.4.9 - Link a button to a device or a memory

Another way of modifying the appearance of a button is to link it to a device or a memory. The button automatically changes aspect according to the state of a device or the value of a memory.

6.4.9.a - link a pushbutton



Display the button properties. You can change the name and link the buttons to a memory or a device. Changes are effective only after the press on the button **Apply**.

Pushbuttons can follow the state of relay, infrared, JPI, X10 rs-232, Midi and EIB. It can also follow the state of memories.

It works as follows: when the condition defined installed is true the button is displayed pressed.

Except the link known as **Icon**. These last link a button with a memory or a device but its display is defined with a list of icons. This button can display 256 different states.

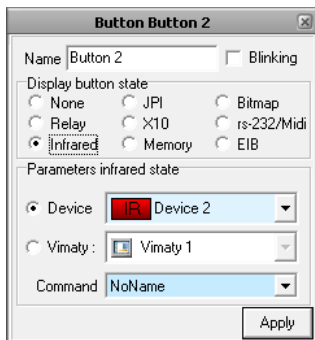
Simple Link - relays, JPI



To link a button to Relay/JPI, select Relay or JPI in dialog box of radio-buttons then in the lower area to choose the relay or the JPI concerned. Click on **Apply**. The button is displayed pressed when the Relay or the JPI is closed, and released when it is opened.

When the blinking checkbox is selected, the button displays alternatively its released state and its pressed state.

Conditional connection - Infrared, X10, Memory, rs-232 and EIB



Select the type of desired link then in the bottom area select a device or a memory. According to the chosen type chosen, it is necessary to specify either a value (memory and EIB) or a command (infrared X10 and rs-232) to define the condition.

In the example, the **Button 2** will be displayed pressed when the device **Device 2** will emit the command **NoName**.

The infrared type offers in addition to the choice between an infrared emits by a control interface or Vimaty.

Link with a list of icons - device or memory



Select the type of **Icon**.

Then in the lower part choose between a memory or a device.

Then finally the desired icons list.

Example :

When the device 1 has value 10 or has emitted orders n°10, the button will display the icon 10 of the list.

When device 1 has value 54 or has emitted command n°54, the button will display icon 54 of the list.

and so on

6.4.9.b - Link a bargraph

A bargraph can be related to control 0/10 volts, a VCA, an analog input, devices EIB of the type 8 bits or with a memory.

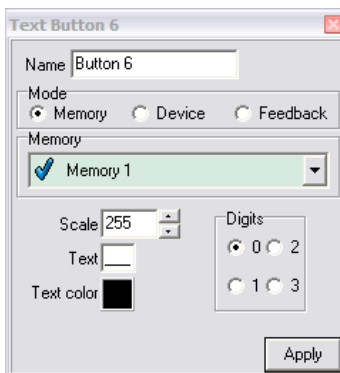


Do like pushbuttons : Select the type then the device or desired memory then validate.

Parameter **Range** defines the number of pixels the cursor can move. In the example opposite, the cursor can move of 258 pixels from its initial position.

The box **Gauge** makes it possible to use the alternative view of the bargraph. In this case it is not any more one cursor which moves on a background image, but an image of size identical to the background image which partially is displayed according to the followed value.

6.4.9.c - Link a dynamic button (text)



Proceed as for the push buttons: Select the type then the device or desired memory then validate.

The difference is that the value is displayed in the form of figure and either icon.

Properties of the texts:

- **Scale** : which value to display if followed element is worth 255
- **Text** : Characters to be displayed after the value
- **Text color** : Define the color of the text
- **Digits** : Define the name of decimal displayed. Vimaty treats only numbers from 0 to 256 but it is possible to include a decimal point in the display. Ex: value 127 can display 127, 12.7, 1.27 or 0.127 following the number of selected decimals.

6.4.9.d - Duplicate a button

After having defined a button with its appearance and its functionalities, it can be useful to duplicate it in order to minimize the change to be made on the following button. Two configurations are possible :

- if the button is copy in the same window as the original : do a drag'n'drop with the button to be copied from the button list to the drawing area, give a name to the new button and apply the necessary modifications to it.

- if the button is copy to another window : select the button, choose **Copy button**, then select destination window and use the menu paste the button.

6.5 - Advanced Properties of Vimaty

6.5.1 - Vimaty macros

Each Vimaty can execute macros according to certain conditions.

6.5.1.a - Initialization of Vimaty

During its power on sequence, Vimaty call macro Initialization. To edit this macro, click on the button **Initialization** under the drawing area. The window of edition of the macros opens.

This macro is useful to initialize the memories with a given value, to position devices in a precise state, etc...

6.5.1.b - The JPI of Vimaty

It is possible to start actions when the Vimaty dry contact is closed and/or opens.

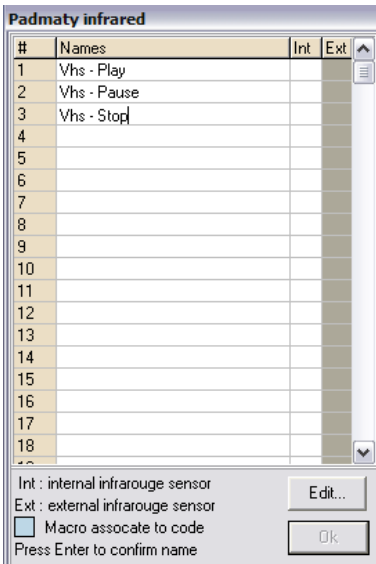
The edition of the macro that must be carried out with the opening of the contact is done while clicking on the button **Open JPI** under the drawing area. For the macro link to close JPI, click on the button **Close JPI**.

6.5.2 - Vimatys infrareds

Vimaty can manage 127 infrared codes. After training of these codes, Vimaty can re-emit them by macro.

Each code can also be use (in condition of having been learned in **Normal** mode) to start macro if it is received by Vimaty either on its internal IR sensor or on its external IR sensor. It is possible to define 2 different macros according to the source from the code IR.

By default Vimaty does not use any infrared codes, it is necessary to declare (to reserve) the codes to be used. For that give a name to the desired code. Click on the button **Infrared** located under the drawing area and the following window opens.



To create macro which must be started with the Vimaty receive an infrared code, select the **Int** box for the internal sensor or **ext** for the external sensor. Click on the **Edit...** button or double click on the box or press on the **F2** key and the edit window of the macros appears.

Put actions in the macro and validate : When the Vimaty will receive the infrared code it call this macro.

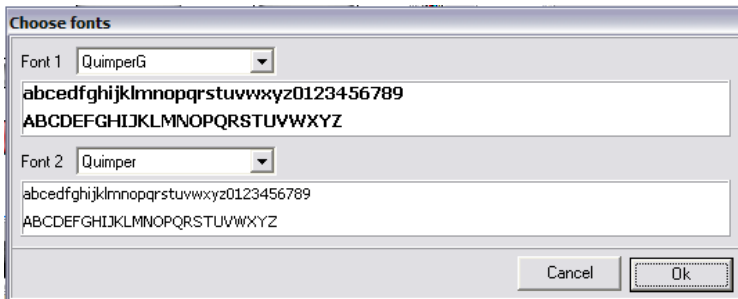
The infrared code of Vimaty must be learned before usage. Only the reserved codes appear in the infrared training part.

To send a Vimaty infrared, use the System command **Send command Infrared** in the macro edition window.

6.5.3 - Vimatys Fonts

The dynamics displays of the text of Vimaty (dynamic Date, hour, temperature and buttons) can use different fonts in order to better adapt to the total display of the application.

To change font, click on the Button F under the drawing area. The window which appears makes it possible to select the 2 fonts that will be used by the application.



Selection of the fonts

Vimaty uses 2 fonts according to the type of element to be displayed. Fonts must be upload in Vimaty during insertion of CompactFlash in Vimaty. Refer to the manual of Vimaty for a description of the method of transfer of the font in Vimaty. The font are copy to CompactFlash during the generation of the project.

7 - THE SYSTEM COMMANDS OF VIMATY

This section describes whole Vimaty commands. Functions marked (70s) works only on Vimaty 70s and 70 EIB.

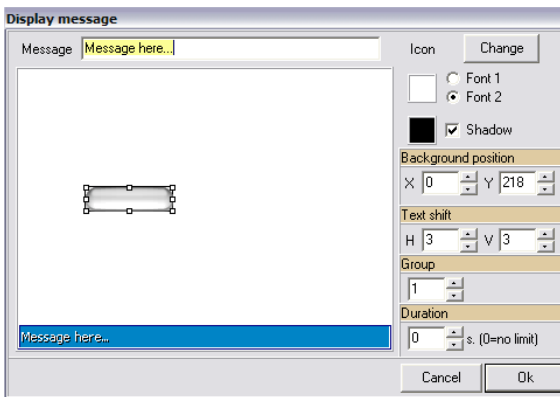
7.1 - Display

7.1.1 - Date-Clock

To switch the display of the date/time to time in a window. So that this function is to be active, the window must have the option display clock or date in its properties.

7.1.2 - Message

To display a message on all Vimaty of a given group. The message is composed of a text and a background image.



Parameters according to can be being modified:

- Background icon
- The font, the color of the text, the color of the shade (existing or not)
- The position of the icon
- The shift of the text compared to the icon.
- The group of Vimaty recipient
- Duration of display of the message.

7.2 - Memory

7.2.1 - Reset memories

Set value 0 for the whole of Vimaty memories.

7.2.2 - Load memories

Load values of the memories of Vimaty from the internal safeguard. Is used with **Save memories**.

7.2.3 - Save memories

Copy the value of each memory in a save field of Vimaty. Is used with **Load memories**.

7.3 - Send command

7.3.1 - Infrared

Make it possible to send an infrared code internal of Vimaty. Pctomaty requires the code to emit as well as the number of screens.

7.3.2 - Fixed string rs-485

Emit a series of byte on the bus rs-485. reserved to advanced users being informed of the protocol rs-485 VITY.

7.3.3 - Memory string rs-485

Same function than the command **Fixed string rs-485** but with the string composed of a series of bytes in Vimaty memory.

7.4 - System

7.4.1 - Else

To add the instruction ELSE in a test in the event of accidental suppression.

7.4.2 - END IF

To add the instruction END IF in a test in the event of accidental suppression.

7.4.3 - Insert comment

Make it possible to add comments in macro. The comments are ignored during the generation of the project.

7.4.4 - Tempo flickering

Make it possible to vary the speed of flickering of the images.

7.5 - Waiting

7.5.1 - Wait (blocking)

Suspends the execution of the program in Vimaty for one duration determined in hundredth of second.

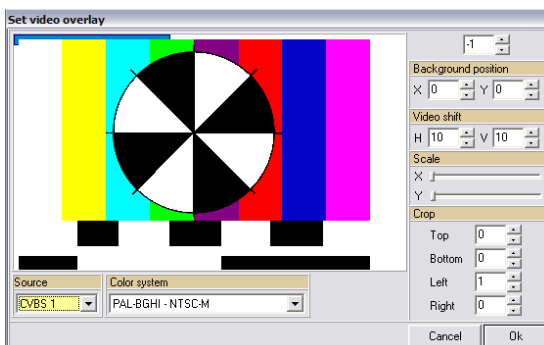
7.5.2 - Wait (not-blocking)

Suspends execution of the macro containing the instruction for a given time. It is possible to start 4 thus "timer". the remainder of the application functions.

7.6 - Vidéo (70s)

7.6.1 - Incrustation (70s) defines

Make it possible to encrust a video window in Vimaty 70s or 70 EIB.



Parameter setting of a video incrustation

For the initialization of the video incrustation, it is necessary to specify the parameters according to:

- Video source (1 of the 4 entries Composite or 1 of 2 entries Y/C or nothing to change):
- Format of the video signal (**System of color**).
- The background icon of the video: for example a framework.
- The position of the icon
- The shift of the video compared to the edge high-left of the icon.
- Compression X and Y of the video
- The trimming of the video in top, low, left & right-hand side

The area representing the screen represents a sight thus indicating the effects of the different parameter of framing.

7.6.2 - Select source

To change the video source of an incrustation without having with all to redefine.

7.7 - Synchronize Vimaty

To synchronize the display of a group of Vimaty. To do that use the name shared name for this window all Vimatys. All the Vimaty of the group concerned that have a window with a share name identical to the one defines in the command, will switch to this window.

7.8 - Copy Value Button

This command is mainly used to know the position of the finger on a bargraph. Copy the position of the finger from the button in a memory. The position is a value ranging between 0 and 255.

For a horizontal bargraph, 0 on the left and 255 on the right.

For a vertical bargraph, 0 bellow and 255 in top.

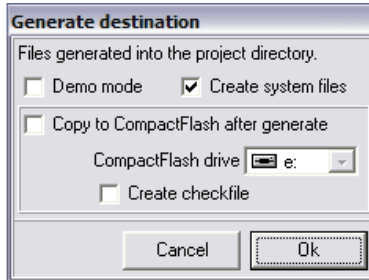
The value copied in the memory defined in the command can then used to be included in rs-232 command with parameter for example. It is also possible to apply operations such as the rule of 3 to adjust the value with the protocol of the device recipient.

8 - GENERATE A PROJECT FOR VIMATY

To transfer a project into Vimaty, it is first of all necessary "to generate it" or "to compile".

Click on the button **Generate selected Vimaty** at the bottom right of Pctomaty, then define the generate option. Generate is always made on hard disk and then be copy to CompactFlash automatically or manually.

From Pctomaty version 2, the transfer of the files is made by comparison between the contents of CompactFlash and the contents of the hard disk in order to copy only files having been modified and this to reduce the transfer time from CompactFlash. The first generation of the project on CompactFlash will copy all the files and will be longest. Then compilations will be shorter.



Compilations parameters

Compilations parameter

- **Demo Mode** : do Vimaty program which does not send an rs-485 command.
- **Create system files** : Generate the system files necessary to reach the "Setup" of Vimaty. Refer to the Vimaty handbook for the use of Setup.
- **Copy to CompactFlash after generate** : Copy files to CompactFlash at the end of compilation. In this case it is necessary to specify in which reader CompactFlash is. If this option is not selected, the files are generated on the hard disk in the repertory of the project. They should then be copied by using the Windows explorer from CompactFlash.
- **Create checkfile** : Between two compilations, Pctomaty compares the generated files. It may be that you are not on origin of CompactFlash. In this case it is possible to format it with format FAT16 with Windows or to ask Pctomaty to generate the file of comparison of CompactFlash before making the choice of the files be copied.

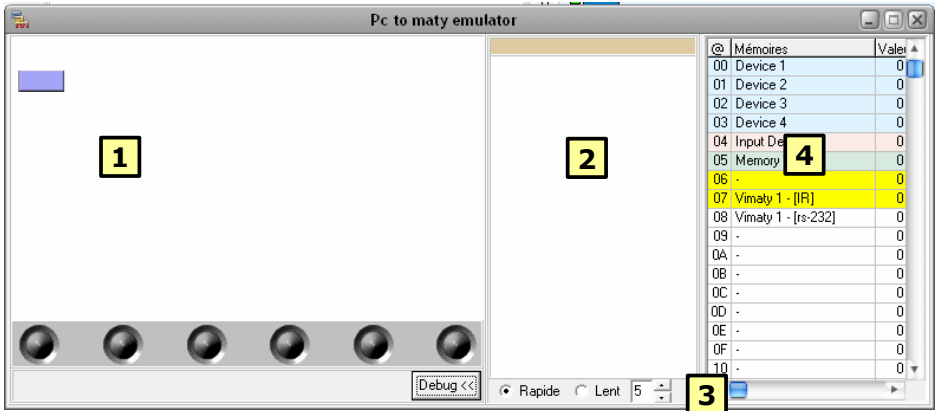
Attention :

You always must check if it remains enough of place on CompactFlash to be able to contain the project.

At the end of the file transfer, put CompactFlash in Vimaty and power on.

9 - THE VIMATY EMULATOR

Pctomaty has a Vimaty emulator. To use it, select Vimaty to be emulated in the list of Vimaty and click on the button Emulator of main toolbar. The next window opens:



- 1** Represent the screen of Vimaty.
- 2** Display the macro in progress of execution
- 3** Setup the speed of execution of the macros
- 4** Display the memories of the devices, and the memories users as well as the values which they contain.

Then open part **2 3 4**, click on button **Debug >>**.

10 - THE PADMATY

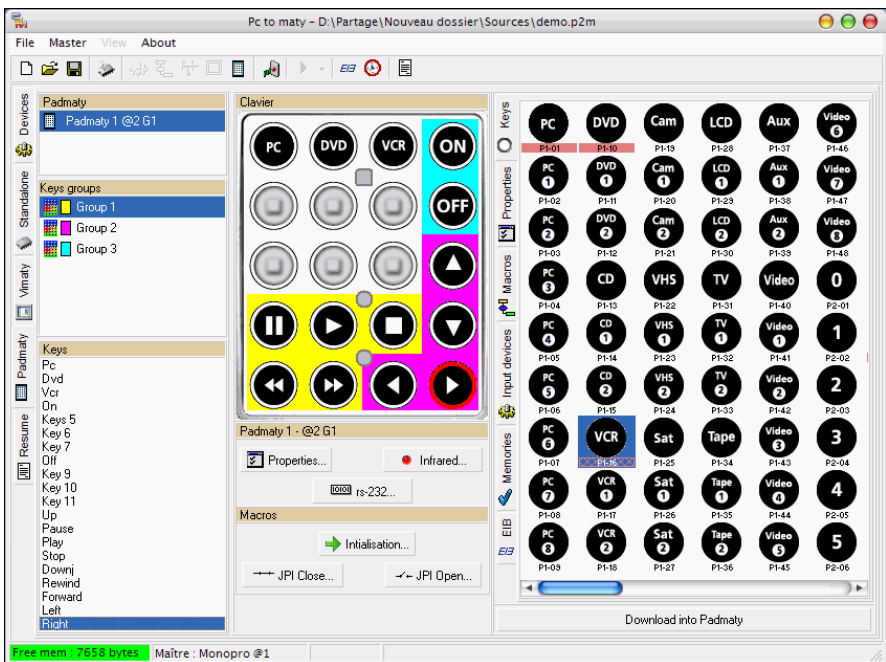
10.1 - Introduction to Padmaty

The Padmaty is a keyboard 20 user-definable keys with the following functionalities :

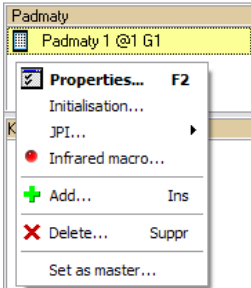
- 8 Programming mode per key
- 3 Macros (press, maintained and release) by key and mode (either 24 macro per key)
- Each key can be illuminated by programming, change of the value of report and the change of state of device and this according to the mode running of the key.
- The whole of the functionalities of Vimaty are usable in Padmaty except the graphic functions.

10.2 - Padmaty page anatomy

Press the Padmaty tab



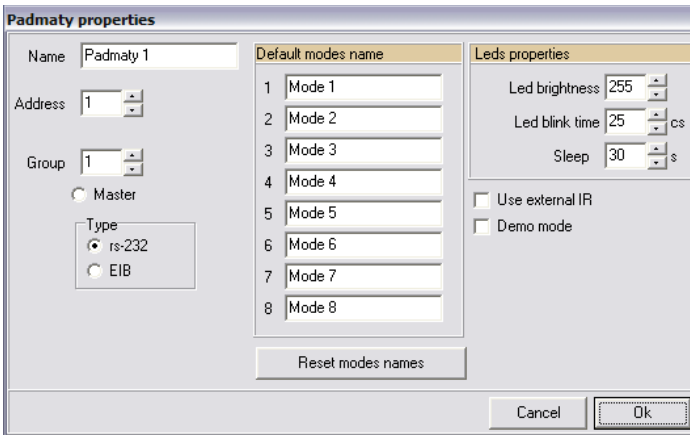
10.3 - Declare a Padmaty



To create a Padmaty keyboard, click right on the Padmaty window located on the left of the main window and choose Add.

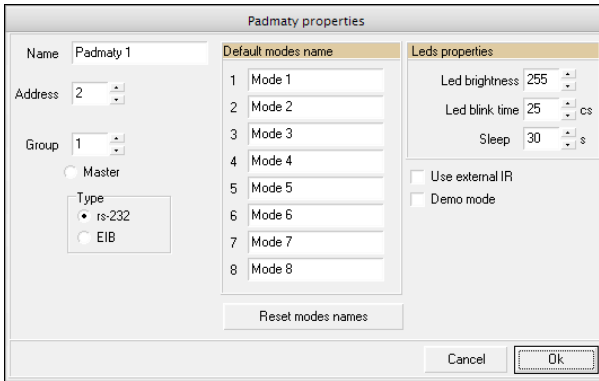
You can also click on the ninth icon of the main toolbar.

The following dialog box appears to define the characteristics of Padmaty.



10.4 - Padmaty properties

When create a Padmaty or when calling its properties, the following window appear



Padmaty properties window

Properties of Padmaty are the same as Vimaty properties for address, group, rs-2322 or EIB, External IR, sleep and demo mode.

Special properties of Padmaty are :

- **Led brightness** : Can change the lightness of backlight
- **Led blink time** : change the blinking speed of keys backlight when keys blinks
- **Modes name** : to make easier to manage Padmaty modes during development, it's possible to give a name to different modes. Ex : mode vcr, dvd etc...

10.5 - Configure the keyboard to help programming

To help programming several thing can be done :

- Put icons onto keys. These icons represent the physical keys available to customize the Padmaty.
- Give a name to each keys.
- Create group of keys : keys of a group will change from one mode to another by programming in only one command.