MapBasic Tutorial
Copyright Notice

This publication is a copyrighted work owned by TYDAC AG, Monbijoustr. 95, 3007 Bern.

No part of this publication or the software provided herewith may be modified, reproduced, transmitted, stored in a retrieval system, translated into any language or computer language, in any form or by any means including electronic, mechanical, magnetic, optical or otherwise dealt with or used except as set out in TYDAC AG's standard licence agreement.

DISCLAIMER

TYDAC AG MAKES NO REPRESENTATIONS, COVENANTS OR WARRANTIES WITH RESPECT TO THE CONTENTS OF THIS PUBLICATION AND DISCLAIMS ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE.

Further, TYDAC AG reserves the right to revise this publication without obligation to notify any person of such revisions.

Trademark Acknowledgements

MapBasic is a registered trademark of MapInfo Corporation, Troy, NY.

All other trademarks and registered trademarks are the property of their respective owners.

Authors of this edition

Carl Nunn, Aligned Assets UK, 6 Queen Street, Gomshall, Surrey, England, GU5 9LY

Hans-Jörg Stark, TYDAC AG, Monbijoustr. 95, 3007 Bern, Schweiz

© 1996 by TYDAC AG All rights reserved.
Writer/Editor: Hans-Jörg Stark
# Table of Contents

**First steps...**

**Introducing MapBasic**

- How simple MapBasic can be... .................................................................9
- Program Structure .....................................................................................9

**Menus**

- Creating a Menu ........................................................................................11
- Adding a Menu to the MapInfo Menu bar .................................................11
- Adding and Removing Menu Items ..........................................................12
- Altering Menu Items .................................................................................13
- Menu Item Syntax .....................................................................................13
- Defining Keyboard Shortcuts .................................................................14
- Help Messages ..........................................................................................15
- Menu Bar Specifics ...................................................................................15
  - Menu Bar Show | Hide...................................................................15
  - The WinSpecific Menu ..................................................................16
  - Create Menu Bar As Default .........................................................16

**ButtonPads**

- Button Types .............................................................................................19
- Creating ButtonPads .................................................................................20
- Alter ButtonPad ........................................................................................20
- Alter Button ..............................................................................................21
- Button Definition Syntax ..........................................................................21
- Custom ToolButtons .................................................................................22

**Dialog Boxes**

- Dialog Control Types ..............................................................................25
  - StaticText ............................................................................................25
  - EditText ..............................................................................................26
Updating objects... .........................................................................52
Things you should know about SQL and update statements .........52
Dear MapBasic User

This little manual shall help you doing your first steps using MapBasic. It shall give you a brief yet compact introduction how to use MapBasic to customize your MapInfo applications.

Since all of the tasks are as different as our clients such an introduction can never be complete. Yet we tried to make it as comprehensive as possible. Therefore we dare hope that you will enjoy this introduction and that the added examples will be of help for you to not only improve your skills but to as well widen your horizon as far as mapping applications are concerned. May this little book be of help to you for a successful and joyful programming in the wide world of maps.

A basic knowledge of MapInfo is assumed whereas programming experience with other programming languages is no prerequisite.

The tutorial is based on MapBasic 4.0 on Windows.

Have lots of fun!

Hans-Jörg Stark
TYDAC AG
How simple MapBasic can be...

MapBasic can be very simple. Take the following program for instance:

Note "Hello Goodbye"

Typically, MapBasic programs involve a bit more than this and the next section illustrates the typical setup.

Program Structure

A typical MapBasic program contains several procedures (these are sometimes called sub procedures). To define a procedure, use the Sub...End Sub statement.

Every program must have a Main procedure. Usually, the Main procedure is declared explicitly (through a Sub Main...End Sub statement). When you run a MapBasic program, MapInfo automatically executes the Main procedure. For more details, see Main.

Within the Main procedure, you can use the Call statement to call another procedure.

Most of the statements in your program will be located within a procedure. However, the following statements must be located outside of any procedure definition: Type, Declare Sub, Declare Function, Define, and Global. Also, Include statements are usually located at the top of a program file, outside of any procedure definition.

The following list summarizes the major program elements, and shows the order in which the elements usually appear:

Global-level statements appear at the top of the program:

Include statements (e.g. Include "mapbasic.def")
Type...End Type statements

Declare Sub statements

Declare Function statements

Define statements

Global statements

‘... followed by the Main procedure:

Sub Main

Dim statements

other statements

End Sub

‘... followed by other procedure definitions:

Sub . . .

Dim statements

other statements

End Sub

‘... and custom function definitions:

Function . . .

Dim statements

other statements

End Function
Most MapBasic applications will begin by adding some menu commands to the Main MapInfo Menu. The application then waits for the user to choose a menu item before proceeding.

Creating a Menu

Before adding a new menu to the MapInfo Menu Bar, you must first create it. The syntax to create a menu is:

```
Create Menu newmenuName [ ID menu_id ] As
    menuitem [ID menu_item_id] [HelpMsg help] {Calling
    handler|As menuName}
    [,menuitem . . . ]
```

For example,

```
Create Menu "Hello Goodbye" as
    "Hello" calling sayHello,
    "GoodBye" calling bye
```

sayHello and Bye are subprocedures that will be called when these menu items are chosen.

Adding a Menu to the MapInfo Menu Bar

Once you have created a new menu you may add it the MapInfo Menu Bar, using the following syntax:

```
Alter Menu Bar Add newmenuName
```
For example

```
Alter Menu Bar Add "Hello Goodbye"
```

In some cases it is preferable to add a menu onto an existing menu (as a sub-menu). The **Tools** menu is provided as a standard additional menu to add Mapbasic applications (otherwise the Mapinfo menu bar would be caused to wrap if a number of applications were running at the same time). To add the *Hello Goodbye* menu to the *Tools* menu use the following syntax:

```
Alter Menu "Tools" Add
"Hello Goodbye" as "Hello Goodbye"
```

Note that any menu can be added to any other menu, it is not limited to the *Tools* menu.

---

### Adding and Removing Menu Items

You may elect to add one or more items to an existing menu such as A standard MapInfo Menu or an Menu you have previously created. To do this use the **Alter Menu** statement. The syntax for Alter Menu is:

```
Alter Menu { menuname | ID menu_id }
Add menudef [ , menudef... ]
```

or

```
Alter Menu { menuname | ID menu_id }
Remove { handler | submenuname | ID menu_item_id } [ , { handler | submenuname | ID menu_item_id } ]
```

For example,

```
Alter Menu "Hello Goodbye" Add
"Before you say Goodbye I say Hello" calling sayHello

Alter Menu "File" Add
"Before you say Goodbye I say Hello" calling sayHello

Alter Menu "File" Remove M_FILE_CLOSE
Alter Menu "File" Add M_FILE_CLOSE
```
Altering Menu Items

Once you have created a menu item you may require to alter its value. This is done with the **Alter Menu Item** statement. The syntax is as follows:

```
Alter Menu Item { handler | ID menu_item_id }
{ [ Check | Uncheck ] | [ Enable | Disable ] | [ Text itemname ] | [ Calling handler | As menuname ] }
```

This enables you to change a menu item's text, to enable or disable the item, change the called handler and check or uncheck the item. For example:

```
Alter Menu Item M_FILE_CLOSE
Text "Close a Table"
Alter Menu Item M_FILE_CLOSE_ALL
Disable
```

The result of this is the customer's menu appears in the tools menu.

How you design and organize your menus will depend on the nature of your application. Some users find hierarchical menus difficult to use.

Menu Item Syntax

The syntax for each menu item is:

```
Menutitle [calling Handler | as menuname]
```

Menu items can be enabled or disabled; disabled items appear grayed out. Some menu items are checkable, meaning that the menu can display a check mark next to the item. At any given time, a checkable menu item is either checked or unchecked.

To set the properties of a menu item, include control codes (from the table below)

<table>
<thead>
<tr>
<th>Control code</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>The menu item is initially disabled. Example: &quot;(Close)&quot;</td>
</tr>
</tbody>
</table>
Defining Keyboard Shortcuts

Menu items can have two different types of keyboard shortcuts, which let the user choose menu items through the keyboard rather than by clicking with the mouse.

One type of keyboard shortcut lets the user drop down a menu or choose a menu item by pressing keys. For example, on MapInfo for Windows, the user can press Alt-W to show the Window menu, then press M (or Alt-M) to choose New Map Window. To create this type of keyboard shortcut, include the ampersand character (&) in the newmenuname or menuitem string (e.g. specify "&Map" as the menuitem parameter in the Create Menu statement). Place the ampersand immediately before the character to be used as the shortcut.

The other type of keyboard shortcut allows the user to activate an option without going through the menu at all. If a menu item has a shortcut key sequence of Alt-F5, the user can activate the menu item by pressing Alt-F5. To create this type of shortcut, use the following key sequences. Note: the codes in the following tables must appear at the end of a menu item name.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-)</td>
<td>The menu item is a horizontal separator line; such a menu item cannot have a handler. Example: &quot;(-)&quot;</td>
</tr>
<tr>
<td>($)</td>
<td>This special code represents the File menu's most-recently-used (MRU) list. It may only appear once in the menu system, and it may not be used on a shortcut menu. To eliminate the MRU list from the File menu, either delete this code from MAPINFO.MNU or re-create the File menu by issuing a Create Menu statement.</td>
</tr>
<tr>
<td>(&gt;)</td>
<td>This special code represents the Window menu's list of open windows. It may only appear once in the menu system.</td>
</tr>
<tr>
<td>!</td>
<td>Menu item is checkable, but it is initially unchecked. For Example: &quot;!Confirm Deletions&quot;</td>
</tr>
<tr>
<td>!...^...</td>
<td>If a caret (^) appears within the text string of a checkable menu item, the item toggles between alternate text (e.g. Show... vs. Hide...) instead of toggling between checked and unchecked. The text before the caret appears when the item is &quot;checked&quot;. Example: &quot;!Hide Status Bar^Show Status Bar&quot;.</td>
</tr>
<tr>
<td>!+</td>
<td>Menu item is checkable, and it is initially checked. Example: &quot;!+Confirm Deletions&quot;</td>
</tr>
</tbody>
</table>

Defining Keyboard Shortcuts
**Windows Accelerator Code**

<table>
<thead>
<tr>
<th>Code</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>/W {letter</td>
<td>%number}</td>
</tr>
<tr>
<td>/W# {letter</td>
<td>%number}</td>
</tr>
<tr>
<td>/W@ {letter</td>
<td>%number}</td>
</tr>
<tr>
<td>/W^ {letter</td>
<td>%number}</td>
</tr>
</tbody>
</table>

---

**Help Messages**

If you include the help clause in the menu item definition a short description of that command may appear in the status bar. For Example:

```
Create Menu "Hello Goodbye" as
"Hello" calling sayHello,
HelpMsg "Says Hello"
"GoodBye" calling bye
HelpMsg "Says Goodbye and leaves"
```

---

**Menu Bar Specifics**

There are some extra functions to be aware of which are specific to the MapInfo Menu bar.

**Menu Bar Show | Hide**

This is important to be aware of particularly when using MapInfo runtimes and you can’t work out where the Menus got to. (The default is Hidden with runtimes).
The WinSpecific Menu

Window specific menus refer the menus "Map", "Browse", "Graph", "Layout". You may manipulate these menus using WinSpecific. For example:

- Alter Menu Bar Remove "WinSpecific"

Create Menu Bar As Default

You can use this command if you wish to restore the menu bar to the default. This can be quite rude if there is another Application running as you will drop the applications menu from the Menu Bar.

Examples:

Example 1:

Try to figure out how the MapBasic code should look like for the following Menu:

![Menu Example](image)

When you either choose the submenu Hello or press Ctrl+H a dialogbox should appear saying "Hello". And when you either choose the submenu Exit or press Alt+X the just created Menu MyMenu should be removed from the Menu Bar.

(The MapBasic code of all the examples are found in the appendix)

Example 2:

Remove all the menus from the main Menu Bar except the main menus File and Help. In the File Menu remove all saving and print options. The result should look like the following screen shot:
(The MapBasic code of all the examples are found in the appendix)