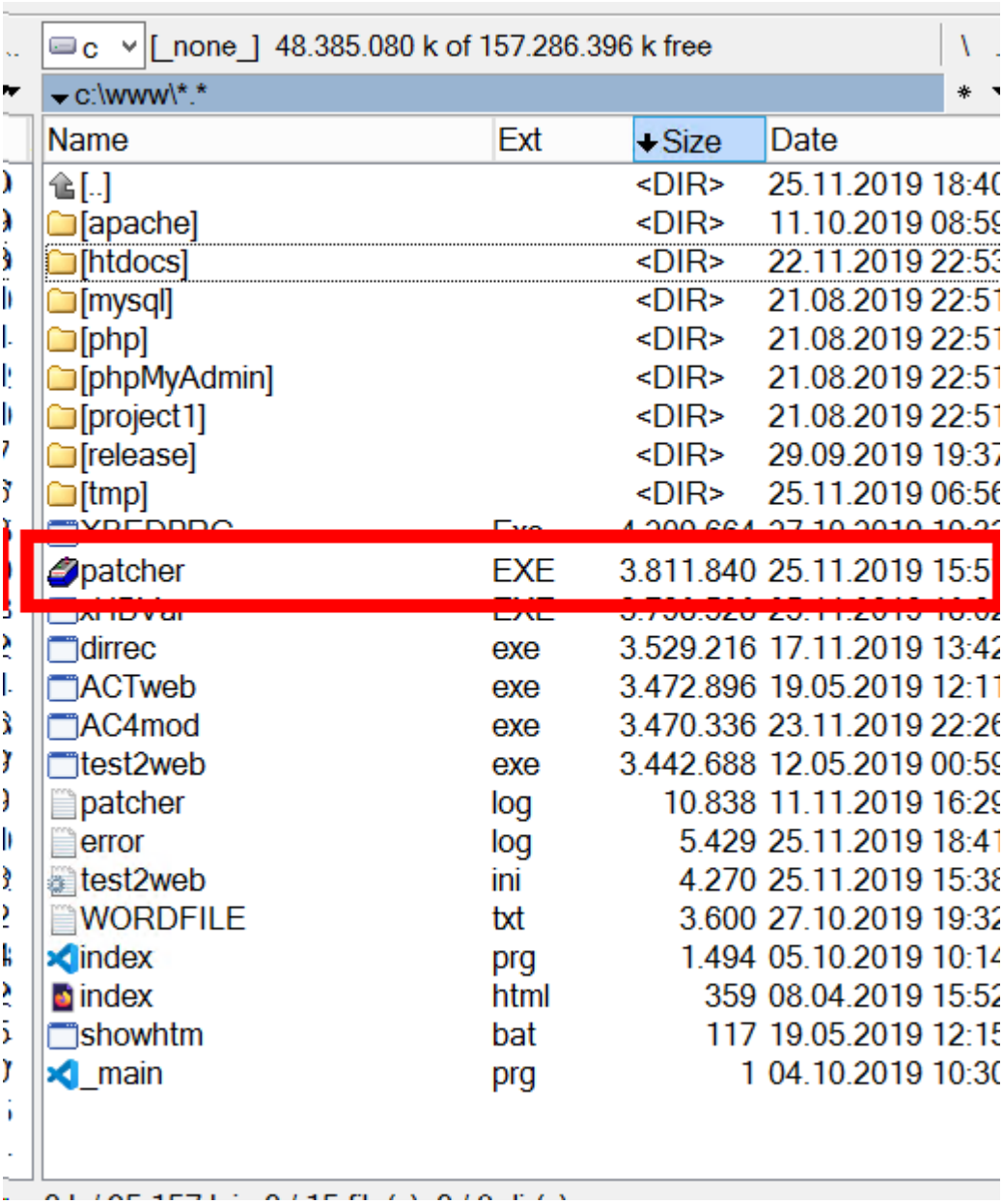
Harbourino is a preprocessor that takes as input a text file written using some programming language syntax
and output another text file following the syntax of another programming language.

Harbourino is a preprocessor that makes-
1) the program easier to develop.
2) easier to read.
3) easier to modify.
The facilities provided by a preprocessor are given below-
1) File inclusion
2) Substitution facility
3) Conditional compilation.

The more I work with Harbourino style the more I think it is revolutionizing also Fivewin programming.
I mean an extra preprocessor and splitting up complexity for "normal" – John Doe programmers is very potential.
For example, for a test, I split up classes and have for every method an own file.
This way you can achieve NASA programming rules: Restrict functions to a single printed page.

HARBOURINO can be used for every kind of project: php, html, prg, etc.

The Harbourino runs as an *.exe on the server in c:\xampp or c:\www



It allows you to split code into several pieces/prgs to take out complexity.

This is especially for beginners and newbies really useful and helps to overcome sticking points at the entry.

It is possible to use mod harbour "include files" also inside an "include file".

mod harbour patcher supports:

siehe: https://www.tutorialspoint.com/cprogramming/c_preprocessors.htm Beschreibung Präpozessor

_main must be in the Harbourino main file

-> link include file

Preprocessor

$-> link include file and preprocesses the file and makes simple textual substitutions - works only at the linked files

|UNIQUEID|
< placeholder > or | placeholder |

use from calling file:

path od only name them path is taken from INI - file

$-> Filename 9999: placeholder=xxxxxxxx;
otherPlaceholder1=yyyyyyyy

you can call files outside with absolute path, too.

You can make subdirectories inside your source code folder and structure this way your include files.

source
EVENTLISTENER
FORM

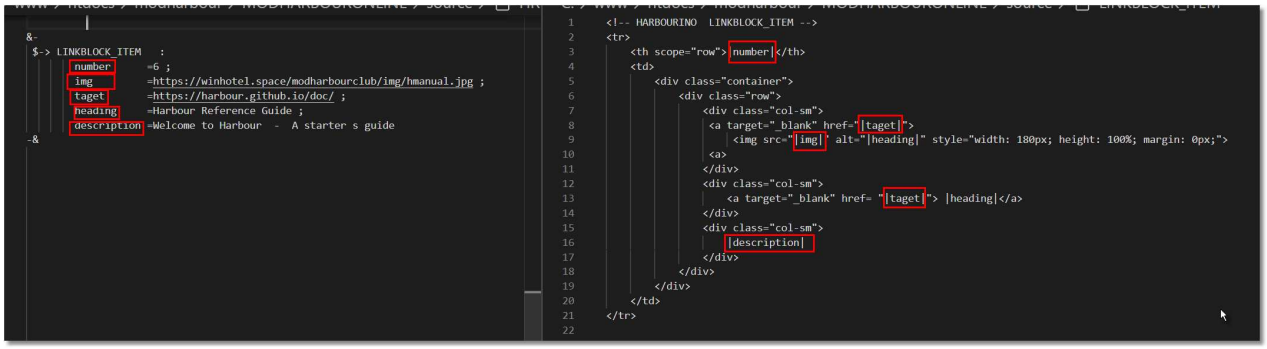
---

preprocessor inside paragraph

Important

```
&-
$-> FORM\INPUT : type=text;
        id = inputUsername;
        labeltext = Email address;
        placeholder = Benutzername;
        name = username;
        text-muted = Use the username "lailton"
-&
```

We can put preprocessor inside paragraph. This way we get nicer code



|- for comments in "include files" - not shown in the "patched" release file.

Paragraph

&- start line

-& end of line

sample for paragraph

```
&-
   |- row wird in tablegebuchtezimmer angehängt

   $(".data-gebuchtTbl tbody").append( "
   <tr data-menge='" + MengeAuswahl  +"' data-beschreibung='"+cBeschreibung+"' data-preis='"+preis+"' data-summe='"+summe+"' data-id='"+id+"'>
       <td  style='visibility:hidden;'>" + id + "</td>
       |- alle 3 Werte sind in einer Spalte
       <td>"+cmenge+"</td><td><b>"+cZiBezeichnung +"</b><br> "+cBeschreibung+"</td>

       <td>"+email+"</td>
       <td><button class='btn btn-info btn-xs btn-deletebooking'>Buchung entfernen</button></td>
```

```
</tr>");

-&
```

is automatically patched to this.

```
$(".data-gebuchtTbl tbody").append( "<tr data-menge='" + MengeAuswahl
 +"' data-beschreibung='"+cBeschreibung+"' data-preis='"+preis+"' data-
summe='"+summe+"' data-id='"+id+"'><td  style='visibility:hidden;'>" +
id + "</td><td>"+cmenge+"</td><td><b>"+cZiBezeichnung +"</
b><br> "+ cBeschreibung+"</td><td>"+email+"</td><td><button
class='btn btn-info btn-xs btn-deletebooking'>Buchung entfernen</
button></td></tr>");
```

HARBOURINO can handle **default values** now

! |clause| value


```
$-> mybutton.prg : prompt= "Konto ändern"; action= ( f_kontoAenderung(), oBrw:refresh() ); ClrBack=METRO_TEAL

$-> mybutton.prg : prompt= "Ende";          action= ( lSave=.T., oDlg:end() )
$-> mybutton.prg : prompt= "Test long Text   F 10 "; action= msginfo("test") ; ClrText=CLR_BLACK; ClrBack=CLR_YELLOW
$-> mybutton.prg : prompt= "Ende default values"; btnheight=60
```

```
|- default values
!|ClrText| CLR_WHITE
!|ClrBack| rgb(2, 117, 216)
!|action| msginfo("action")
!|btnheight| nBtnHeight

nOffSet := If( oBtn == nil, 25, nOffSet + 15 + oBtn:nWidth )
nPixPerChar := GetTextWidth( oDlg:hWnd,|prompt| , Setup():oFontFLATBTN:hFont )  + 30
@ oDlg:nHeight - 70, nOffset BTNBMP oBtn PROMPT |prompt| SIZE nPixPerChar, |btnheight| PIXEL OF oDlg FLAT COLOR |ClrText|, |ClrBack| ACTION |action|
```

Publishing

Publish

For publishing you just have to drag&drop your ***_main.prg** (here MyModTest_Main.prg) in the **MODHarbour Patcher** window. It puts the code together again and in your release-folder you will find the file to publish → MyModTest.prg