An Arduino Debugging Tool, Version 1.0

By R. G. Sparber

Protected by Creative Commons.[1]

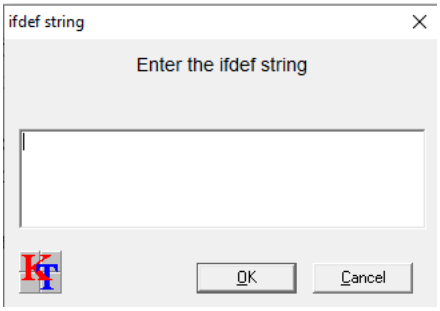
This tool depends on having KeyText, which I use many times each day.

When I debug my Arduino code on a device with a USB, I can print to a monitor. Here is a typical output:

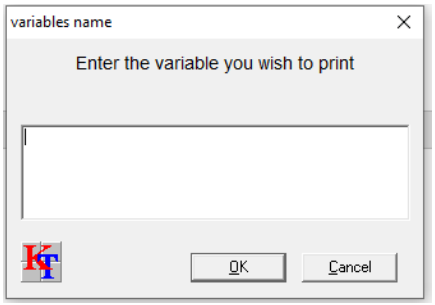
```
loop(): 76.     Time Stamp  5652 ms
InputArray[1] = 99
```

Most of this text is generated by the macro and compiler. It says that this output was in the function `loop()` at line `76`. It has been `5652` milliseconds since power-up. The variable, `InputArray[1]`, equals `99`.

To plant this test code into a place in my program, I call on KeyText.



When I hold down Shift and Control and then press "-", I get my first prompt which asks for the name of a control variable. When this variable has been defined, the remaining code compiles. Otherwise, my test code does not compile, so it does not take up any program space nor use any real-time. Say I enter `print1`.



Then I am prompted for the variable I wish to print. I'll enter `myHatSize`.

[1] This work is licensed under the Creative Commons Attribution 4.0 International License. To view a copy of this license, visit http://creativecommons.org/licenses/by/4.0/ or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

The macro then injects

```
#ifdef print1
Serial.print(__FUNCTION__);
Serial.print(F("(): "));
Serial.print(__LINE__);
Serial.print(F(".    Time Stamp  "));
Serial.print (millis() - StartForTimeStampULong);
Serial.println(F(" ms"));
Serial.print(F("myHatSize = "));
Serial.println(myHatSize);
#endif
```

`#ifdef print1` and `#endif` mark the boundaries of the diagnostic code. Only if I have `#define print1` someplace in my code, will these lines compile. I have as many of these `#define` statements as needed to gain full control over which print blocks I want active at a given time. For example, one of my programs has this at the top of the file:

```
//#define DiagPrint1
//#define DiagPrint2
//#define DiagPrint3
//#define DiagPrint4
#define DiagPrint5
//#define DiagPrint6
//#define DiagPrint7
//#define DiagPrint8
//#define DiagPrint9
```

I have 9 different groups of print blocks, but only `DiagPrint5` is active.

```
#ifdef print1
Serial.print(__FUNCTION__);
Serial.print(F("(): "));
Serial.print(__LINE__);
Serial.print(F(".    Time Stamp  "));
Serial.print (millis() - StartForTimeStampULong);
Serial.println(F(" ms"));
Serial.print(F("myHatSize = "));
Serial.println(myHatSize);
#endif
```

__FUNCTION__ and __LINE__ talk to the compiler, so I get the function's name and line number.

I then print out my timestamp, which is particularly handy when a problem occurs randomly or after a given time. It is also helpful when I am running more than one processor simultaneously, and they trade messages.

Near the top of my file, I have

```
unsigned long StartForTimeStampULong;
```

In `setup()`, I have

```
StartForTimeStampULong = millis();
```

which sets it to the current time. Each time I do a diagnostic print, my timestamp shows the time since `setup()` ran.

The last section prints out the selected variable.

This KeyText macro generates this code does all of the work:

```
{Speed 8}#ifdef {Ask ^"ifdef string" "Enter the ifdef string"}
Serial.print(__FUNCTION__);
Serial.print(F("(): "));
Serial.print(__LINE__);
Serial.print(F(".    Time Stamp  "));
Serial.print (millis() - StartForTimeStampULong);
Serial.println(F(" ms"));
Serial.print(F("{Ask ^"variables name" "Enter the variable you wish to print"} = "));
Serial.println({variables name});
#endif
```

{Speed 8} slows down the printing into the file, so I don't miss any characters.

The macro prints `#ifdef` and then asks for the associated name. After a few more lines of code, the macro prompts for the name of the variable I want to print out. A useful feature of KeyText is the ability to prompt for a variable and then use it in more than one place. I print out the variable's name as text and then print out the contents of this variable.

I welcome your comments and questions.

If you want me to contact you each time I publish an article, email me with "Subscribe" in the subject line. In the body of the email, please tell me if you are interested in metalworking, software plus electronics, kayaking, and/or the Lectric XP eBike so I can put you on the right distribution list.

If you are on a list and have had enough, email me "Unsubscribe" in the subject line. No hard feelings.

Rick Sparber
Rgsparber.ha@gmail.com
Rick.Sparber.org