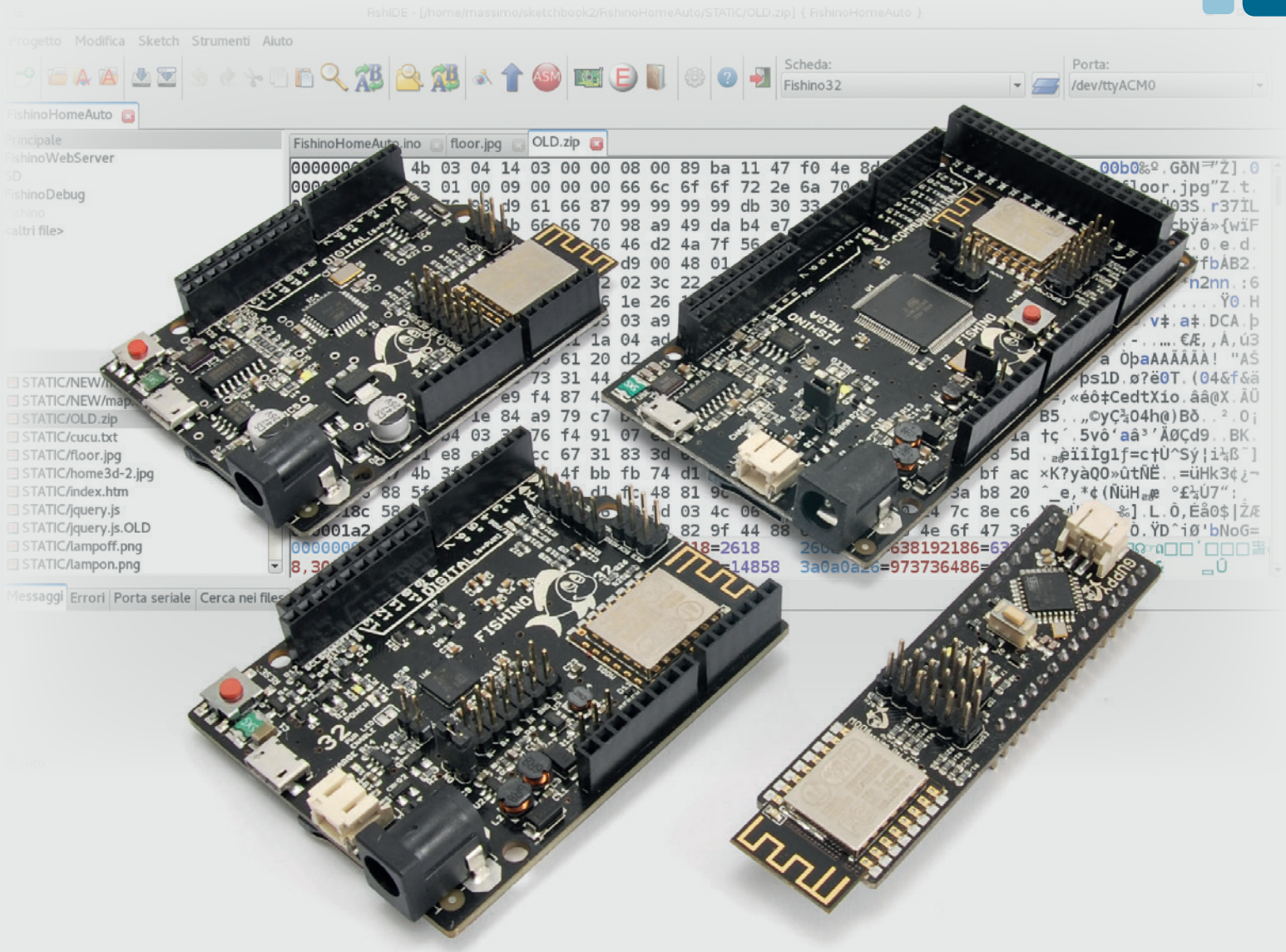

FishIDE

Un nuovo ambiente di sviluppo nato per le nostre schede Fishino, ma adatto anche alle Arduino ufficiali e clone. Seconda ed ultima puntata.

di MASSIMO DEL FEDELE

Il mese scorso ci siamo lasciati dopo aver introdotto e descritto la prima parte del funzionamento e delle utility di FishIDE, l'ambiente di sviluppo da noi creato per le piattaforme Fishino, che contrariamente all'IDE Arduino (rispetto al quale rappresenta comunque un'interessante alternativa anche per le board ufficiali e per i cloni), nasce già predisposto per supportare tutta la famiglia Fishino e il relativo hardware, offrendo qualcosa in più.

Ora proseguiamo e concludiamo la trattazione partendo dall'Editor, che rispetto a quello dell'IDE ufficiale è più interessante e nutrito. Si tratta di uno strumento fondamentale per la creazione di codice destinato a Fishino e perciò molto curato.

L'EDITOR DI FISHIDE

A nostro avviso, una delle funzioni più piacevoli da utilizzare in FishIDE è proprio l'editor di codice,

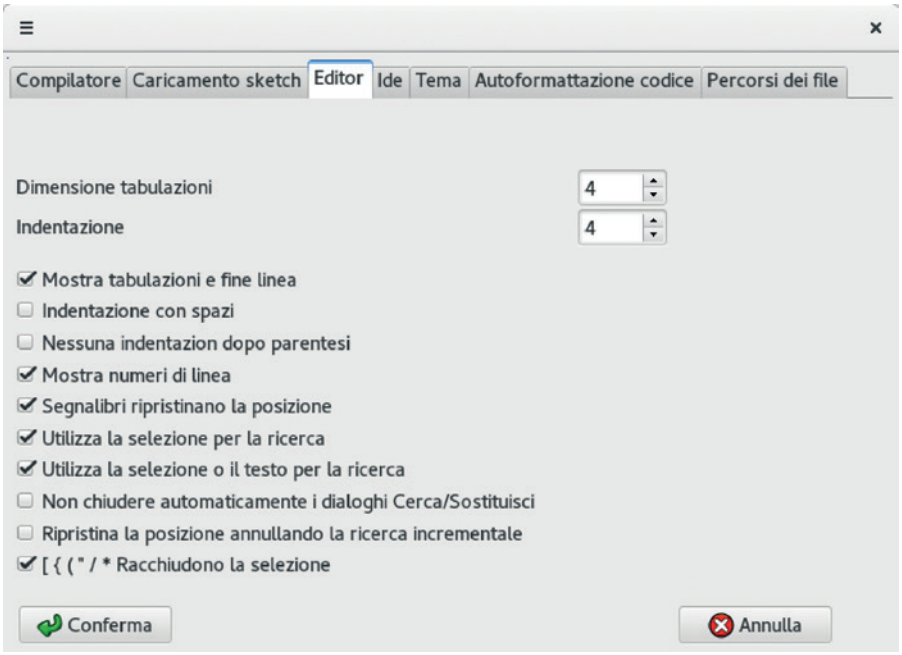

Fig. 1 *- Impostazioni dell'editor.*

che possiede parecchie caratteristiche interessanti. Il funzionamento dell'editor si può personalizzare tramite l'apposita pagina delle impostazioni mostrata in **Fig. 1**. Di seguito descriviamo le sezioni corrispondenti.

- **Dimensione tabulazioni**: quest'opzione imposta la dimensione dei caratteri di tabulazione, inseribili facendo clic sul tasto Tab o inseriti automaticamente grazie all'indentazione automatica del codice.
- **Indentazione**: quest'opzione imposta il numero di spazi utilizzati nell'indentazione del codice, ovvero il rientro di blocchi di codice, per esempio, all'interno di un'istruzione while. Suggeriamo di impostare questa opzione come multiplo della dimensione tabulazioni; il valore predefinito (4) consente una visibilità ottimale dei rientri nel codice.
- **Mostra tabulazioni e fine linea**: quest'opzione, quando attivata, rende visibili i caratteri di tabulazione e fine linea; è molto comoda per visualizzare spazi non desiderati e/o la corretta indentazione del codice.
- **Indentazione con spazi**: attivando questa opzione, i rientri del codice avvengono inserendo un numero di spazi corrispondente al valore 'Indentazione' visto sopra; disattivandola, l'indentazione viene effettuata tramite caratteri di tabulazione. Consigliamo di lasciarla disattivata.
- **Nessuna indentazione dopo parentesi**: normalmente andando a capo dopo una parentesi, il cursore si porta verso destra di un numero di posizioni pari al valore 'Indentazione' visto sopra, in modo da facilitare la scrittura ordinata del codice. Selezionando questa casella si disabilita tale funzionalità.
- **Mostra numeri di linea**: come si può vedere dalle immagini che abbiamo inserito finora, questa opzione mostra i numeri di linea sulla sinistra della finestra dell'editor; disattivandola i numeri spariscono.
- **Segnalibri ripristinano la posizione**: questa opzione è riservata ai segnalibri, non ancora abilitati nella versione corrente. È stata inserita in previsione di un'aggiunta futura.
- **Utilizza la selezione per la ricerca**: questa opzione, se abilitata, fa in modo che il testo selezionato venga usato automaticamente quando si esegue una ricerca; disattivandola occorre inserire il testo

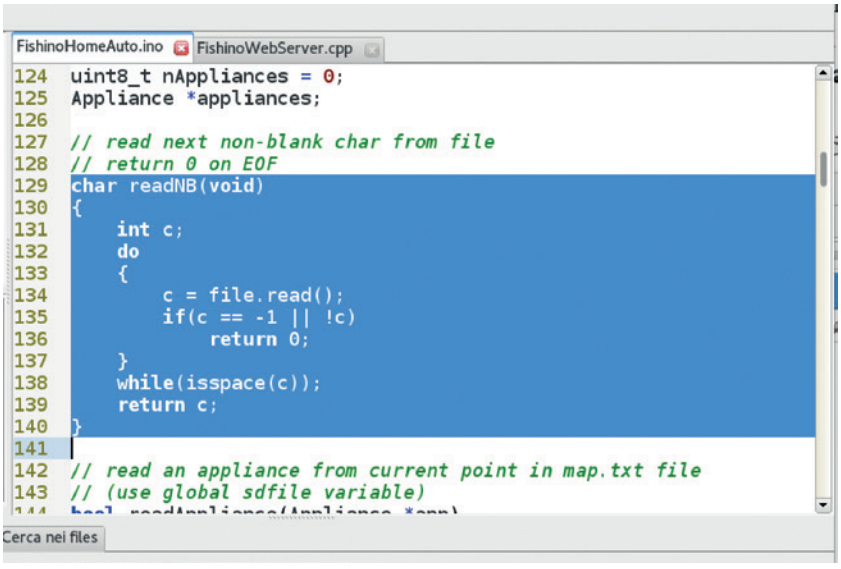

Fig. 2 *- Selezione del blocco di codice.*

manualmente nel campo di ricerca.

- **Non chiudere automaticamente i dialog box cerca/sostituisci**: come vedremo a breve, le funzioni cerca/sostituisci (nel file corrente, non su più file) aprono una o più finestre di dialogo direttamente nella parte inferiore della finestra dell'editor. Normalmente cliccando nel testo la finestra di dialogo viene chiusa automaticamente; abilitando questa opzione la finestra resta aperta.
- **Ripristina la posizione annullando la ricerca incrementale**: i dialog box di ricerca hanno un'opzione per la ricerca incrementale, ovvero, man-mano che si inseriscono i caratteri della parola da cercare il cursore nel testo si sposta alla prima parola trovata corrispondente. Abilitando questa opzione, chiudendo la finestra di dialogo il cursore torna al punto di partenza, mentre lasciandola disattivata il cursore rimane sul punto in cui era alla chiusura della finestra di dialogo.
- **[{("/* racchiudono la selezione**: questa è una delle particolarità più interessanti di FishIDE, e la vedremo in dettaglio in seguito. Abilitando questa opzione, se abbiamo selezionato del testo nell'editor, facendo clic su uno dei caratteri elencati il testo viene racchiuso da una coppia di caratteri corrispondenti. Ad esempio, selezionando la parola PIPPO e cliccando sulle virgolette otterremo "PIPPO".

Torniamo al nostro sketch *FishinoHomeAuto.ino* e proviamo, ad esempio, a commentare un blocco di codice; per far questo occorre selezionare il blocco con il mouse (posizionandosi ad inizio blocco, premendo e tenendo premuto il tasto sinistro e trascinando a fine blocco prima di rilasciarlo) o con la tastiera (posizionandosi ad inizio blocco, premendo e mantenendo premuto il tasto Maiusc e spostandosi con le frecce a fine blocco, come in **Fig. 2**.
Premiamo ora il tasto '*' per commentare il codice con un commento multilinea /*.....*/ (**Fig. 3**).
Come potete vedere, il testo viene circondato dai caratteri /* e */, quindi commentato. Ripremiamo '*' senza modificare la selezione per decommentarlo e tornare al testo originale; sempre senza modificare la selezione, premiamo ora '/', ottenendo una serie di commenti a linea singola (**Fig. 4**).
Ripremiamo il carattare '/' per tornare al testo originale. Come si può facilmente intuire, pre-

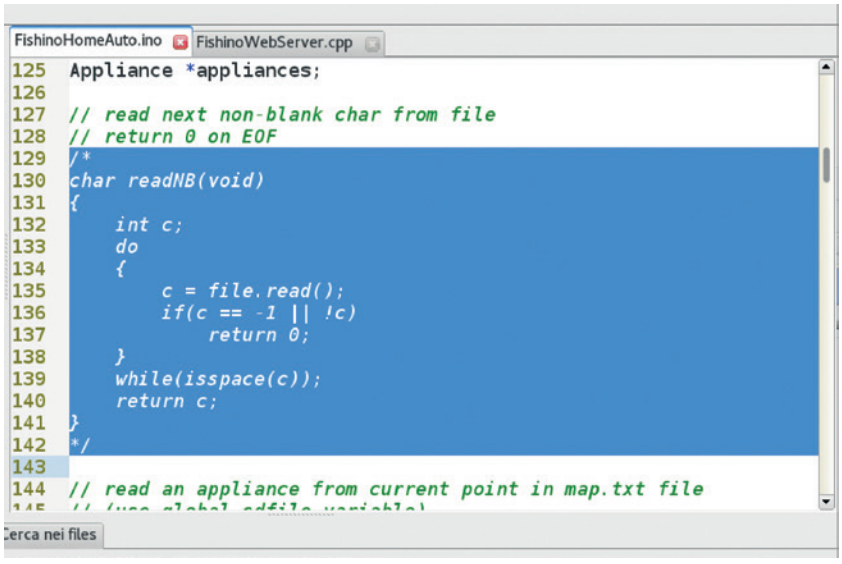

***Fig. 3** - Commento del codice selezionato.*

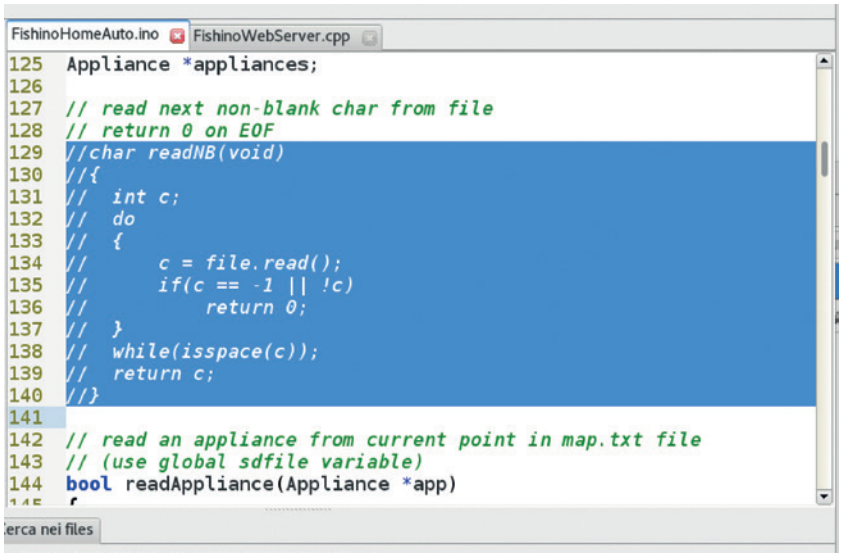

***Fig. 4** - Commenti a linea singola.*

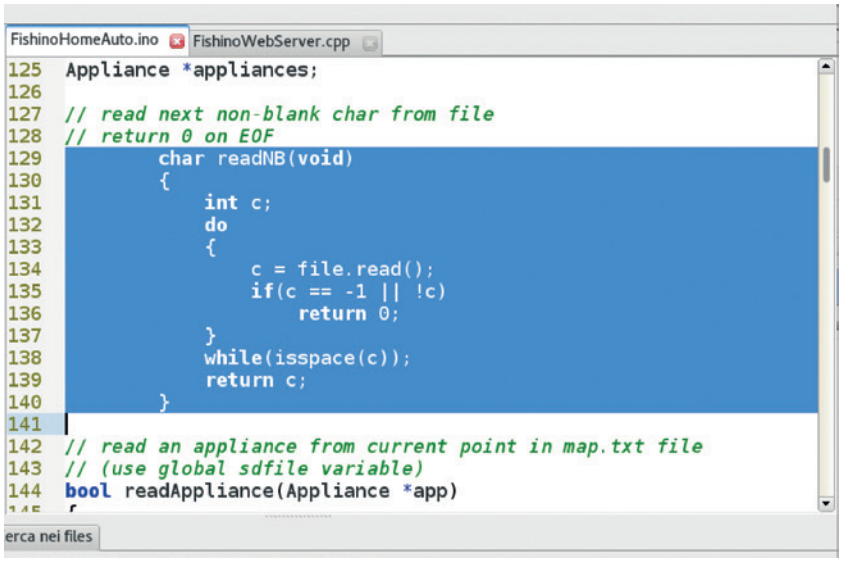

Fig. 5 *- Spostamento dell'indentazione.*

Fig. 6 *- Pulsanti di editing.*

mendo le virgolette su un testo, questo viene circondato dalle medesime, e lo stesso avviene con una parentesi aperta di qualsiasi tipo, facilitando enormemente la digitazione.
Per spostare verso destra un blocco di codice basta premere Tab; sempre con la selezione precedente, premiamo due volte Tab e, come si nota in **Fig. 5**, il blocco di codice si è spostato verso destra di due posizioni di indentazione (4+4 caratteri, o meglio, 1+1 tabulazioni); premendo Maiusc+Tab altre due volte, il blocco torna al suo posto.

RICERCA E SOSTITUZIONE

Abbiamo voluto dotare il nostro FishIDE di un buon numero di funzioni per la ricerca e la sostituzione di testi nei file, accessibili tramite questi quattro pulsanti, corrispondenti ad altrettante voci del menu Modifica oppure alcuni shortcut da tastiera:
Vediamo in dettaglio ogni singolo comando:

Ricerca nel file corrente; apre il menu di ricerca in basso nella finestra dell'editor;

Fig. 7 *- Funzione Ricerca.*

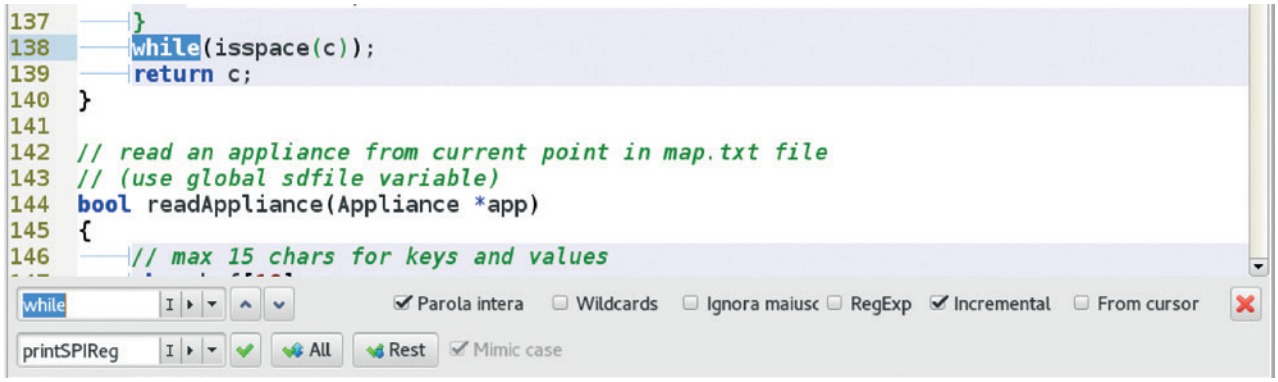

Fig. 8 *- Finestra di dialogo del comando Sostituisci.*

Sostituisci nel file corrente; apre il menu di ricerca/sostituzione nella finestra dell'editor;

Ricerca nei file; apre un dialogo separato per cercare del testo in file multipli;

Sostituisci nei file; apre un dialogo separato per ricerca e sostituzione in file multipli.

I primi due pulsanti agiscono sul file corrente, ovvero su quello aperto nella finestra dell'editor; apriamo per esempio il menu Ricerca (**Fig. 7**).
In questo caso abbiamo digitato la parola while nel campo di ricerca, abilitando le opzioni Parola intera e Ignora maiusc; il cursore si è posizionato sulla prima parola corrispondente; l'opzione Incremental fa si che il cursore si sposti man mano che digitiamo. Cliccando sulle due frecce il cursore si sposta alla parola trovata precedente o successiva.

Fig. 9 *- Finestra del comando di ricerca.*

sostituire la prima. I pulsanti All e Rest permettono di sostituire tutte le parole trovate e quelle trovate dal punto corrente in poi, rispettivamente.
Usare con cautela, anche se l'editor dispone della possibilità di annullare qualsiasi comando tramite undo!
I comandi di ricerca e sostituzione nei file sono molto potenti (ed il secondo anche pericoloso!) permettendo di operare su file multipli in base a criteri specifici.
Vediamo il comando di ricerca nei file che riporta varie opzioni, come visibile nella (**Fig. 9**).
Facendo clic sul pulsante OK viene avviata la ricerca, ed i risultati mostrati nel pannello inferiore (finestre ausiliarie) nel tab 'Cerca nei file' che verrà attivato automaticamente (**Fig. 10**).
Cliccando su ogni linea verrà aperto il file corrispondente nell'editor e il cursore verrà posizionato sulla parola trovata. Il comando "sostituisci nei file" apre la seguente schermata proposta dalla **Fig. 11**.
Premendo "OK" viene avviata la ricerca e la sostituzione, in questo caso della parola "while" con la parola "WHILE" (potete provarlo, poi fare la sostituzione inversa per non trovarvi con un codice inutilizzabile!). A differenza del comando "ricerca nei file" nella sostituzione vengono aperti TUTTI i file dove viene eseguita la sostituzione stessa, quindi vi troverete facilmente con decine di tab aperti.
Come sempre, tutti gli editor

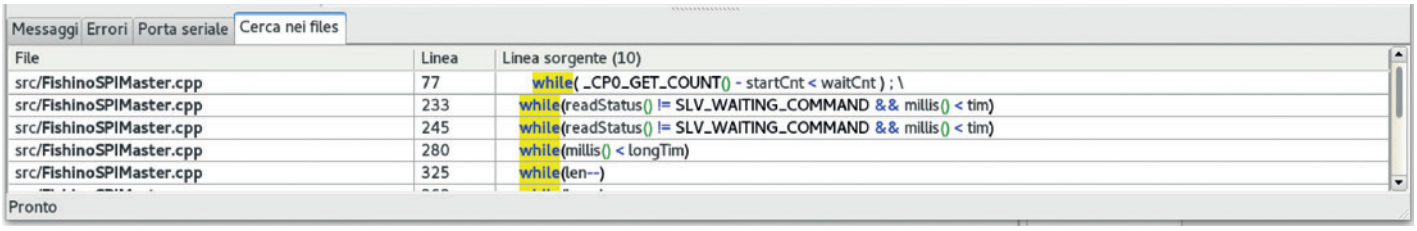

Fig. 10 *- Risultato della ricerca nei file.*

Le opzioni sono moltissime e per spiegarle tutte ci vorrebbe troppo tempo; è molto più semplice se le provate!
Il comando Sostituisci apre un dialog box analogo (**Fig. 8**).
Qui abbiamo due linee, la prima delle quali contiene la parola da cercare con le opzioni identiche alle precedenti, mentre la seconda riporta la parola che andrà a

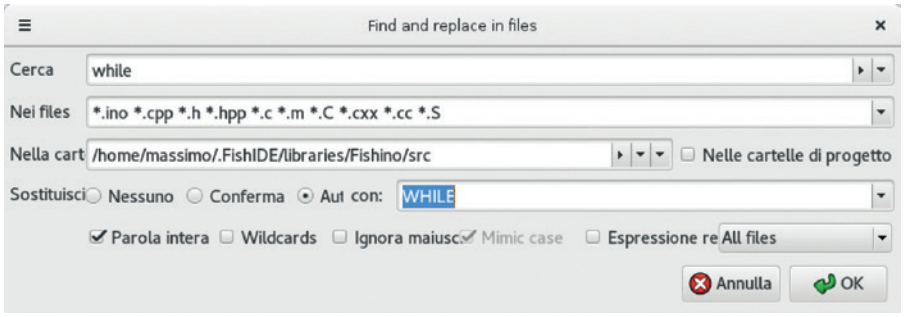

Fig. 11 *- Finestra di dialogo "sostituisci nel file".*

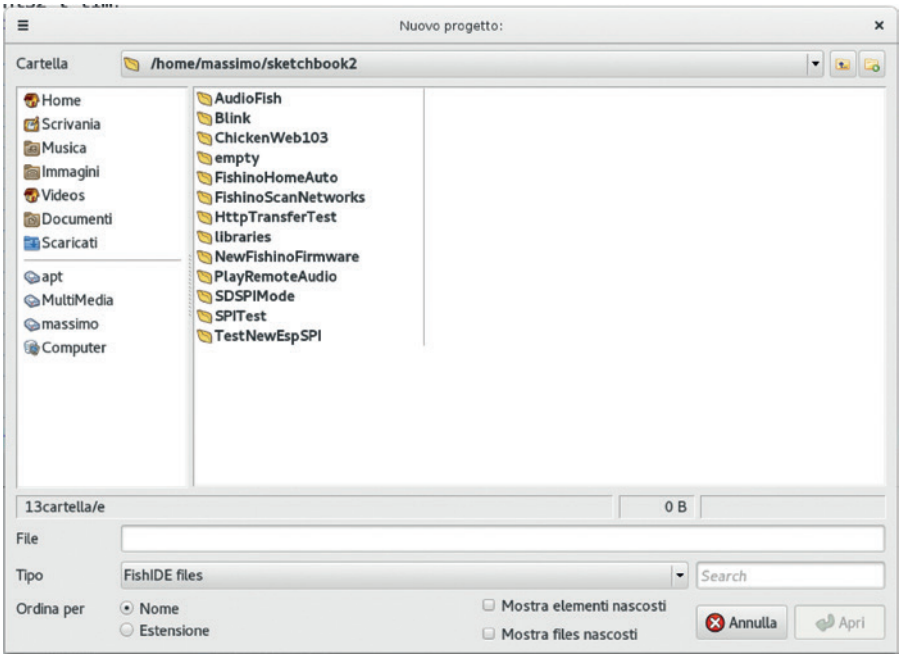

Fig. 12 *- Finestra per la creazione di un nuovo progetto.*

dispongono di un "undo" che permette di annullare le modifiche, ma occorre farlo finestra per finestra, cosa piuttosto lunga, quindi... attenzione all'uso che fate del comando!
La sostituzione nei file risulta comodissima, ad esempio, per cambiare nome ad una classe in un progetto o in una libreria, oppure per sostituire in toto un tipo dati con un altro.

CREAZIONE DI UN NUOVO SKETCH

La creazione di un nuovo sketch viene eseguita facendo clic sul pulsante "Nuovo progetto"  o impartendo il comando attraverso la relativa voce del menu; si aprirà, quindi, la finestra di dialogo di selezione del file, riportante il contenuto della cartella contenente i nostri sketch (**Fig. 12**).

Occorre quindi inserire il nome dello sketch (senza estensioni né altro) per crearne lo "scheletro". La **Fig. 13** riporta quel che appare dopo aver creato lo sketch 'Prova', che verrà aperto.

AGGIUNTA DI UNA LIBRERIA ALLO SKETCH

Aggiungiamo ora una libreria al nostro sketch; potremmo semplicemente inserire una #include in testa ma, sfruttando le funzionalità di FishIDE la cosa è ancora più semplice!
È sufficiente puntare con il mouse nell'**AREA LIBRERIE** (il riquadro in alto a sinistra, ricordate?) e premere il tasto destro del mouse, per far apparire un piccolo menu (**Fig. 14**).
Selezioniamo quindi la voce '**Aprire una libreria esistente ed aggiungerla allo sketch**', dopodiché apparirà la finestra di dialogo mostrata in **Fig. 15**.
Qui potete notare la suddivisione delle librerie nelle tre categorie descritte nella prima puntata, ovvero **Piattaforma**, **Sistema** e **Utente**; selezioniamo la categoria Utente e dentro questa la libreria

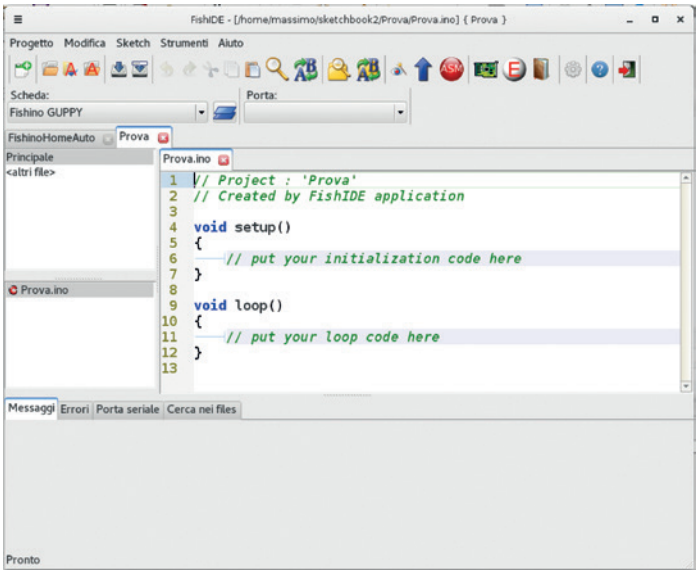

Fig. 13 *- Lo sketch appena creato.*

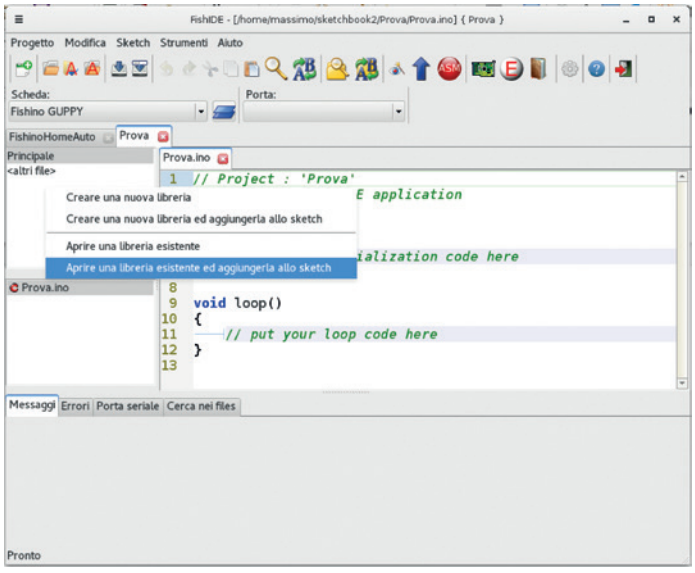

Fig. 14 *- Il menu per l'aggiunta delle librerie.*

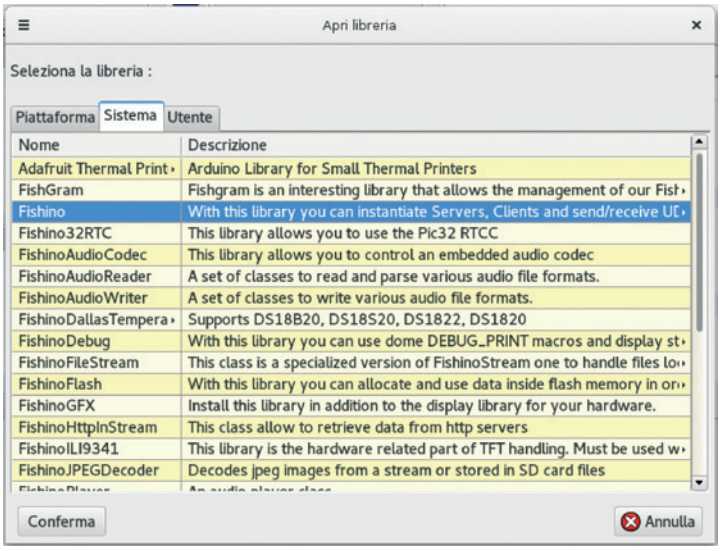

Fig. 15 *- Suddivisione delle librerie.*

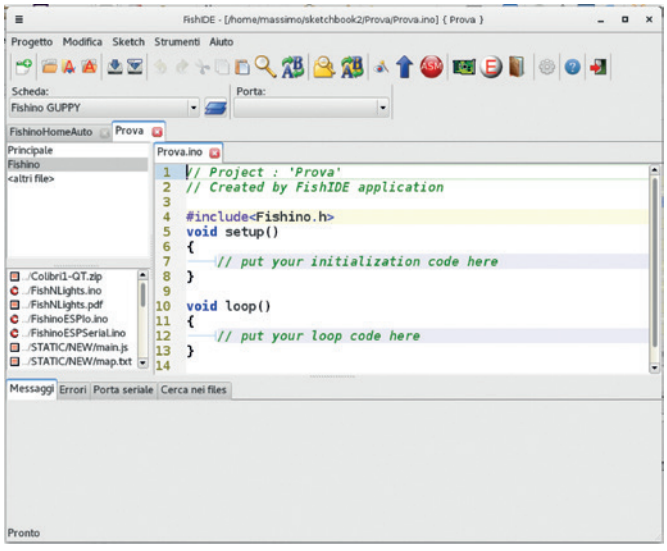

Fig. 16 *- Comparsa della libreria nel riquadro.*

Fishino e premiamo conferma. La libreria apparirà nel riquadro **AREA LIBRERIE** e contemporaneamente nel nostro sketch verrà inserito **#include<Fishino.h>** (**Fig. 16**).

AGGIUNTA DI UN FILE ALLO SKETCH

Per aggiungere un file al nostro sketch (per esempio un file include, che chiameremo "Prova.h") per prima cosa nel riquadro **AREA LIBRERIE** selezioniamo la prima voce, "**Principale**", in modo che nel riquadro sotto, **AREA FILE**, apparirà l'elenco dei file del nostro sketch; in quest'ultima area premiamo il tasto destro del mouse per far apparire il menu contestuale, nel quale sceglieremo la voce "Aggiungi file al progetto" (**Fig. 17**).
Ci apparirà la consueta finestra di dialogo per la selezione dei file, già posizionato nella cartella dello sketch, nel quale occorrerà solo inserire il nome del file, completo di estensione **.ino**.
Selezionando 'Apri' il file verrà aggiunto e direttamente aperto nell' IDE (**Fig. 18**).
Come potete notare, nel caso di file sorgente (.h, .cpp o .ino) FishIDE inserisce uno scheletro di codice per velocizzarvi le operazioni; lo scheletro è ovviamente generico e potrebbe non andare bene. Nel caso dei file .h viene creata automaticamente una classe con lo stesso nome del file, e con alcune funzioni predefinite al suo interno. Questa funzionalità è disattivabile in un'apposita pagina del dialogo "impostazioni".

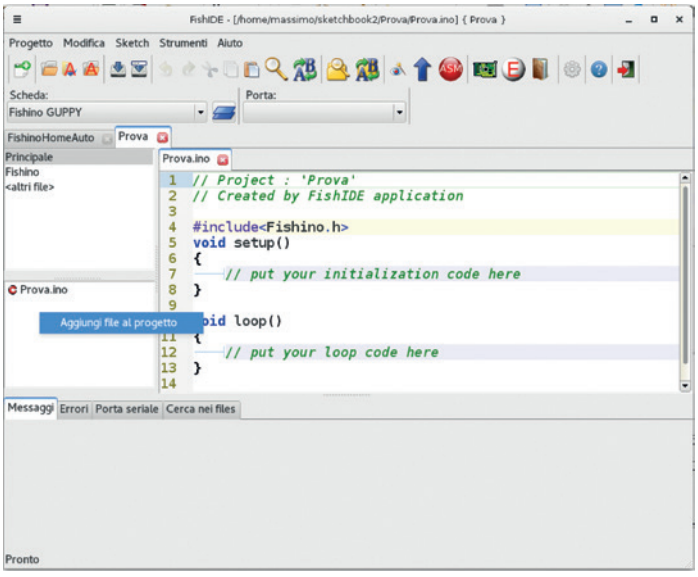

Fig. 17 *- Scelta del comando di aggiunta file.*

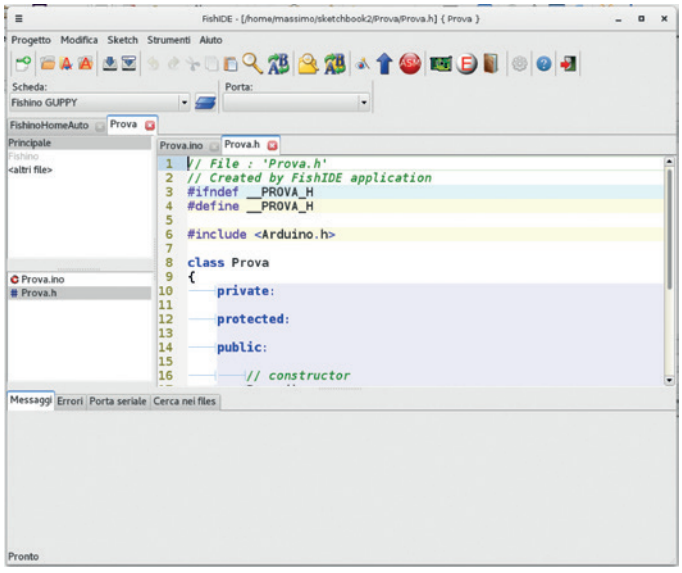

Fig. 18 *- Il file aggiunto e aperto nell'IDE.*

Fig. 19 *- Contenuto del file library.properties.*

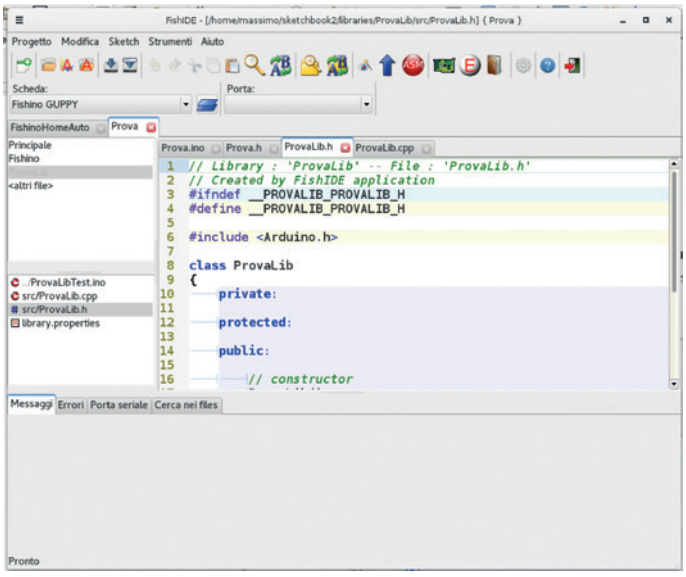

Fig. 20 *- Scheletro dello sketch d'esempio.*

CREAZIONE DI UNA LIBRERIA

FishIDE permette la creazione di nuove librerie in modo molto semplice ed immediato; la creazione e scrittura di librerie, infatti, è stato il principale motivo per cui abbiamo scritto la nostra IDE alternativa.

È sufficiente posizionarsi nella solita AREA LIBRERIE, premere il tasto destro del mouse e selezionare la voce "Creare una nuova libreria" oppure "Creare una nuova libreria ed aggiungerla allo sketch", a seconda delle preferenze. Apparirà una finestra di dialogo particolare, che corrisponde al contenuto del file 'library.properties' delle librerie (**Fig. 19**).

È possibile anche creare direttamente lo scheletro di uno sketch di esempio d'uso, lasciando attiva la casella in basso a sinistra. Facendo clic su Conferma vengono generati tutti i file e le directory necessarie per la nostra libreria (**Fig. 20**).

È poi possibile aggiungere file alla libreria, cancellarne, modificarli, eccetera, esattamente come si fa per gli sketch, solo selezionando la libreria nell' AREA LIBRERIE invece della voce Principale.

MODIFICARE L'ASPETTO DELL' IDE

Questa possibilità potrebbe sembrare superflua, ma per molti ritrovarsi in un ambiente di sviluppo "familiare" come colori e fonts potrebbe essere indispensabile; a chi scrive, ad esempio, non piacciono i temi "dark" nelle interfacce, che invece altri prediligono.

Quasi tutti i colori e i font dell'interfaccia sono modificabili tramite la pagina che si apre cliccando sulla scheda "Tema" accessibile delle impostazioni e le sue sottopagine (**Fig. 21**).

Le modifiche non sono immediate (purtroppo il tempo per lo sviluppo di questa funzionalità era ridotto, quindi abbiamo preferito inserirla, anche se un po' complessa da utilizzare, piuttosto che farne a meno); le modifiche comunque non comportano "danni" e si può sempre riattivare il tema principale.

Alcune modifiche (colori, alcuni fonts) appaiono subito correttamente premendo il tasto "conferma", altre (spaziature nel cambio di font e altro) verranno visualizzate in modo approssimativo fino alla chiusura e riapertura dell' IDE.

Essendo una funzionalità "cosmetica", lasciamo a voi il piacere di sperimentare!

Nel caso poi realizziate un tema particolarmente ben fatto ce lo potrete inviare (i temi possono essere salvati ed esportati) in modo da poterlo inserire direttamente in una prossima versione.

VISUALIZZAZIONE DEL CODICE DISASSEMBLATO

FishIDE, come l'IDE di Arduino, non dispone di un debugger incorporato, spesso utile per trovare gli errori nel codice. C'è da dire che un debugger richiede un hardware aggiuntivo, solitamente, e delle schede predisposte.

La scheda Fishino32, ad esempio, è predisposta per il debugging tramite il connettore ICSP ma, a meno di non utilizzare gli strumenti di sviluppo della Microchip (PicKit3 e MPLAB IDE) la cosa è pressochè inutile.

Molto più utile invece risulta la possibilità di avere il cosiddetto "stack dump" in caso di crash dello sketch, cosa prevista nella serie a 32 bit delle schede Fishino. Nella serie ad 8 bit non è possibile per motivi insiti nel controller utilizzato.

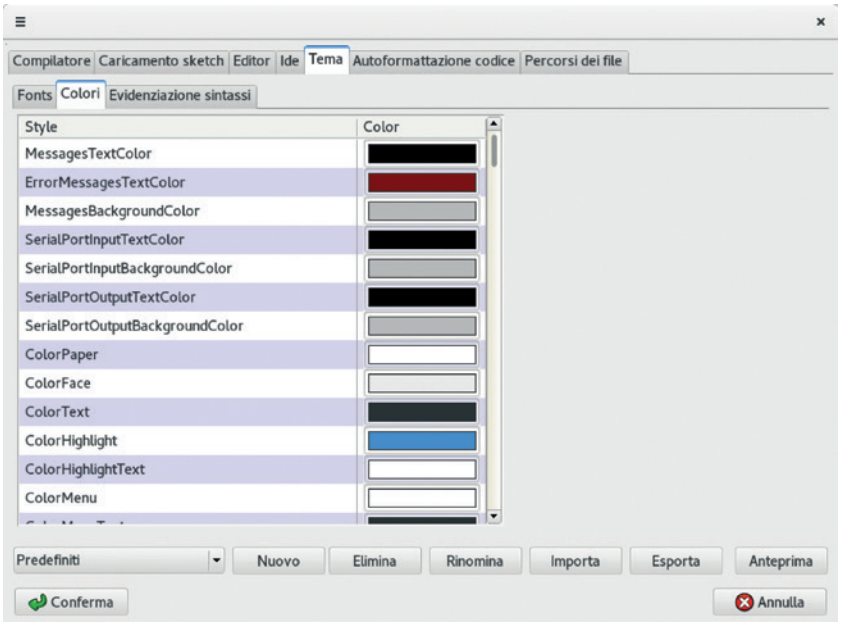

Fig. 21 *- Pagina Tema.*

Cos'è uno stack dump? È semplicemente un elenco di indirizzi, in esadecimale, delle ultime funzioni chiamate nello sketch (o nelle librerie) prima del crash, a partire da quella che ha causato l'errore andando a ritroso. Per la precisione, gli indirizzi sono quelli successivi alle chiamate alle funzioni, ovvero il punto esatto dopo la chiamata.
Ma cosa ce ne facciamo di un elenco di indirizzi ? Normalmente nulla, a meno, ovviamente, di non disporre di qualche strumento che dagli indirizzi permetta di risalire alle linee del codice sorgente!
In FishIDE, al momento (la cosa è in via di ulteriore sviluppo!) abbiamo previsto la possibilità di avere un disassembly dello sketch, ovvero un listato del codice macchina caricato nel controller, inframmezzato con le linee di codice sorgente. Con questo, ed avendo a disposizione gli indirizzi presenti in uno stack dump, è possibile risalire alle linee dello sketch o delle librerie immediatamente precedenti al crash, e quindi individuare facilmente l'errore.
Il listato si ottiene cliccando sul pulsante di disassembly  , che farà aprire una pagina nell'editor contenente il listato.
È sufficiente quindi prendere un indirizzo dello stack dump e cercarlo con la funzione ricerca, per risalire alla linea dello sketch corrispondente. Nella **Fig. 22**, ad esempio, vi mostriamo lo sketch StackDumpTest contenuto tra gli esempi della libreria FishinoDebug, che potrete provare (disponendo di una Fishino32 o della nuova FishinoPiranha).
Non spaventatevi, perché sebbene la cosa sembri complicata, una volta provata si rivelerà uno strumento indispensabile per la ricerca di errori difficili da trovare!

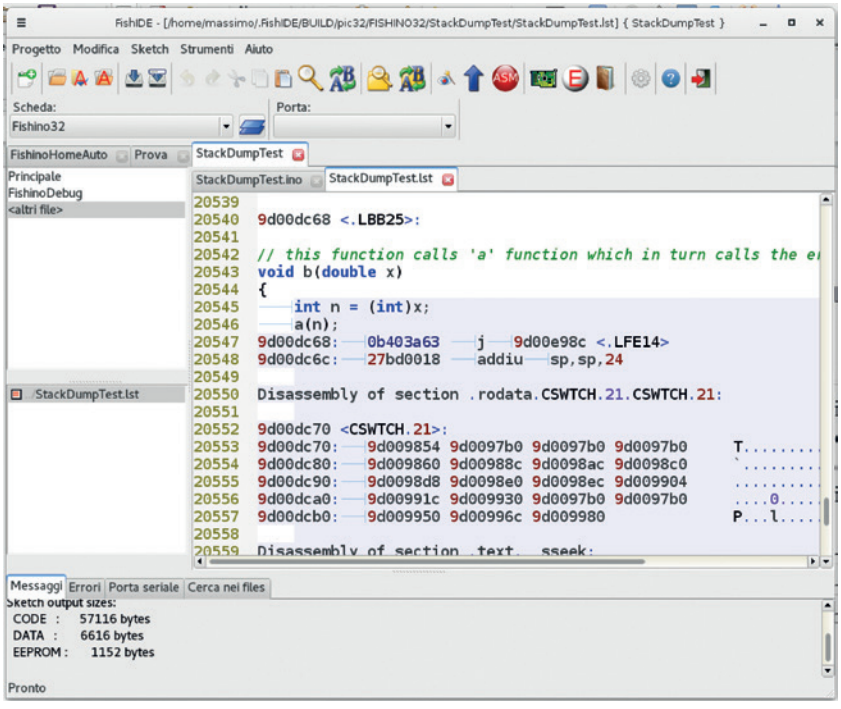

Fig. 22 *- Debug del file.*

CONCLUSIONI

Ebbene, siamo arrivati alla fine della presentazione di FishIDE; vi abbiamo introdotto e insegnato a utilizzare questo prezioso strumento di sviluppo, che, lo ricordiamo, è scaricabile dal sito *www.fishino.it* nella sezione Download.
Da ora avete un alleato in più nella realizzazione di applicazioni firmware per i vostri progetti, potendo contare su un IDE più completo di quello Arduino e che supporta nativamente tutte le board Fishino.