StormC

COLLABORATORS			
	TITLE : StormC		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		July 25, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

StormC

1.1 StormC.guide

```
Anteprima StormC

Software e documentazione
(c) 1995 / 96 by HAAGE & PARTNER COMPUTER GmbH

Tavola dei contenuti

  Licenza d'uso

 Capitolo  1   Benvenuto ad una nuova era
 Capitolo  2   Configurazione minima
 Capitolo  3   Installazione
 Capitolo  4   Cosa fare in caso di problemi "insolubili"
 Capitolo  5   Corso
 Capitolo  6   Il tuo primo programma
 Capitolo  7   Generazione di un nuovo progetto
 Capitolo  8   Make e le dipendenze tra i moduli
 Capitolo  9   Modifica e creazione di un file sorgente
 Capitolo 10   Compilazione
 Capitolo 11   Esecuzione di un programma compilato
 Capitolo 12   Il debugger
 Capitolo 13   Sezioni di un progetto
 Capitolo 14   Caratteristiche dello StormC
 Capitolo 15   Menu dei comandi

 Diritti d'autore

 Modulo d'ordine
```

1.2 StormC.guide/ST_Ordine

```
Per favore stampa il seguente modulo con la tua stampante e invialo
compilato per lettera o FAX, oppure via email.

Segna i prodotti desiderati e indica chiaramente in stampatello
```

i tuoi dati completi di P.IVA se desideri fattura.

Indirizzalo a:

C.A.T.M.U snc
Casella Postale 63

I-10023 Chieri (TO)

Fax: 011 9415237
email: solotre@mbox.vol.it{ub}

MODULO ORDINE (indica gli articoli desiderati)

* Mandatemi la versione completa dello StormC al prezzo di 619000 lire.

* Desidero effettuare un aggiornamento dal mio vecchio sistema di sviluppo "commerciale", allego i dischi originali dell'ultima versione da me in possesso:

  Nome linguaggio: _____

  Numero di serie: _____

  al prezzo di lire 417000. Effettuerete la spedizione dopo aver ricevuto il pacchetto con i dischi come dichiarato sopra.

* in aggiunta gradirei ricevere in omaggio il CD-ROM di IPISA '95 (se ancora disponibile).

I prezzi sono comprensivi di IVA (16%), spese postali e rimborso contrassegno.

* Richiedo spedizione URGENTE e pagherò per questo una maggiorazione di lire 10.000 all'importo del prodotto.

Cognome e Nome: _____

Società: _____

Indirizzo: _____n. _____

CAP: _____ Citta': _____

Provincia: _____

Telefono: _____ Fax: _____

E-mail: _____

Data: _____ Firma: _____

1.3 StormC.guide/ST_DirittiDAutore

Diritti d'autore, marchi registrati e brevetti:

Commodore e Amiga sono marchi registrati di proprietà ESCOM Inc.
SAS e SAS / C sono marchi registrati di proprietà SAS institute.
Amiga, AmigaDOS, Kickstart and Workbench sono marchi registrati
di proprietà ESCOM Inc.
StormC/C++, StormWizard sono di proprietà esclusiva della
HAAGE & PARTNER COMPUTER Ltd.

I prodotti non di proprietà della HAAGE & PARTNER COMPUTER Ltd. sono citati esclusivamente per informare il lettore e non rappresentano alcuna violazione di diritti.

1.4 StormC.guide/ST_Licenza

Licenza d'uso e contratto relativo

1 In generale

(1) Oggetto di questo contratto è l'utilizzo di programmi per computer della HAAGE & PARTNER COMPUTER GmbH, inclusi i manuali ed altro materiale, sia esso scritto o registrato, ad essi pertinente anche ricevuto successivamente.

(2) La società HAAGE & PARTNER COMPUTER GmbH. e/o il distributore indicato nel prodotto sono proprietari di tutti i diritti relativi ai prodotti ed ai marchi.

2 Diritto di usufrutto

(1) L'acquirente riceve il diritto, non trasferibile e non esclusivo, di usare il prodotto acquistato su un singolo computer.

(2) In aggiunta l'utilizzatore può realizzare una sola copia di sicurezza.

(3) L'acquirente non è autorizzato a gettare o distruggere il prodotto acquistato, ad affittarlo, ad offrire sub-licenze o di metterlo in altri modi a disposizione di altre persone.

(4) È vietato cambiare il prodotto, modificarlo o riassemblarlo. Tale divieto riguarda anche il cambiamento, il "reverse engeneering" ed il riutilizzo delle parti.

3 Garanzia

(1) La HAAGE & PARTNER COMPUTER GmbH garantisce che fino al momento della consegna i supporti sono fisicamente privi di difetti materiali e di fabbricazione e che il prodotto può essere utilizzato come descritto nella documentazione.

(2) Difetti del prodotto consegnato sono rimossi dal fornitore entro un periodo di garanzia di sei mesi dalla consegna. Questo avviene attraverso la sostituzione gratuita o sotto forma di un aggiornamento, a discrezione del fornitore.

(3) La HAAGE & PARTNER COMPUTER GmbH non garantisce che il prodotto sia conforme alle aspettative del cliente.
La HAAGE & PARTNER COMPUTER GmbH non si assume alcuna responsabilità per possibili danni causati dall'uso di questo prodotto.

(4) L'utente è avvisato che con la tecnologia attuale non è possibile realizzare software privo di difetti.

4 Altro

(1) In questo contratto tutti i diritti e le responsabilità delle parti contraetenti sono espressi esplicitamente. Non esistono altri accordi. Sono valide soltanto modifiche per iscritto, accettate e firmate da entrambe le parti contraenti.

(2) Per ogni contestazione relativa a questo contratto il foro competente è quello della HAAGE & PARTNER COMPUTER GmbH.

(3) Se alcune clausole del presente contratto dovessero essere difformi rispetto alla legislazione locale, o divenirlo in seguito a circostanze successive, o se dovessero apparire situazioni non regolate dal presente contratto, tutte le clausole non invalidate rimarranno comunque in vigore. In sostituzione o in aggiunta alla clausole valide, dovrà essere formulato un accordo fra le parti che approssimi il più possibile, nel rispetto della legislazione vigente, l'intendimento delle parti quando hanno sottoscritto il presente contratto.

(4) Qualsiasi violazione della licenza di questo accordo o del copyright e dei diritti relativi ai marchi sarà perseguita a termini di legge.

(5) L'installazione del prodotto costituisce espressione dell'accettazione delle condizioni di questa licenza.

(6) Qualora l'acquirente non concordi con il presente accordo di licenza, dovrà immediatamente restituire il prodotto al fornitore.

Settembre 1995

Nota alla traduzione italiana del contratto di licenza:
la traduzione in lingua italiana del contratto di licenza ha valore soltanto informativo; per tutti gli effetti di legge, rimandiamo

alla versione originale in lingua inglese fornita con il prodotto.

Si ricorda che in Italia dal 1 Gennaio 1993 sono in vigore delle leggi per la tutela del software (DPR n.518 del 29 Dicembre 1992), che prevedono una pena da tre mesi a tre anni di reclusione e la multa da Lit.500.000 a Lit. 6.000.000.

1.5 StormC.guide/ST_Benvenuto

Benvenuto in una nuova era della programmazione Amiga.

Con l'acclusa versione DIMOSTRATIVA del nostro nuovissimo compilatore conoscerai le capacità di un linguaggio di programmazione innovativo.

In un ambiente completamente integrato troverai tutto ciò che necessiti per la programmazione. Cuore e centro del sistema è la semplicità del gestore dei progetti, dal quale tutti gli altri componenti sono richiamati e controllati.

Il gestore dei progetti non è semplicemente un migliore MAKE, ma un vero gestore dei moduli del tuo progamma. Fra di essi, per esempio, sono gestiti: font, librerie oggetto, documentazione, script ARexx, immagini e risorse. Anche il compilatore, l'editor e le opzioni di progetto sono gestiti da qui.

Se hai l'impressione che controllare tutto ciò sia troppo complicato posso rassicurarti con quello che seguirà. Guarda le prossime pagine e con il primo esempio descritto capirai molto velocemente che tutto può essere controllato molto facilmente ed intuitivamente.

Un ulteriore componente del sistema è l'editor con la sua particolare abilità di enfatizzare le parole chiave e la sintassi con diversi colori. Con la colorazione del testo puoi leggere il tuo programma molto più facilmente, in quanto noterai meglio la sua struttura.
Oltre a questo ti aiuta ad evitare errori mentre scrivi o modifichi i tuoi programmi: non appena scrivi una parola chiave o il nome di una funzione Amiga, la parola sarà colorata e saprai di averla scritta correttamente.

Ora lascia che ti presenti lo straordinario debugger. Straordinario perché non c'è differenza tra editor e debugger: il debugger è integrato con l'editor, questo ti permette di leggere il programma in esecuzione come se lo stessi scrivendo utilizzando tutte le caratteristiche dell'editor. Puoi inserire breakpoint, cercare funzioni e variabili, con la facilità offerta dall'editor e la colorazione automatica del sorgente semplificherà anche il tuo lavoro di individuazione e correzione degli errori.

Quando abbiamo progettato le caratteristiche dello StormC avevamo delle idee che non sono ancora state implementate, ma che aggiungeremo in futuro. Ad esempio, prossimamente sarà possibile modificare il programma dal debugger mentre il programma stesso è in esecuzione. Questo ti consentirà di non abbandonare il debugger compilando e rilanciando lo stesso programma. Come puoi immaginare questo semplificherà e velocizzerà lo sviluppo del software.

Altro grande aiuto al debugger è il nostro "RunShell": con esso è possibile individuare errori tipici di programmazione relativi al sistema operativo più velocemente; un esempio di errore comune e frequente può essere sintetizzato con le funzioni AllocMem() e FreeMem(). L'allocazione della memoria è facile da fare, ma la restituzione al sistema operativo sembra essere un problema per molti programmatori: o dimenticano di liberare tutta la memoria o ne liberano solo una parte, causando eccezioni a livello di microprocessore (guru meditation).

Il "RunShell" traccia tutti i dati principali di uso delle risorse di sistema in quanto conosce le funzioni usate per tali fini e rileva anche quando sono utilizzate con argomenti non corretti.

Grande vantaggio di "RunShell" è la possibilità di lanciare il debugger in qualsiasi momento, anche mentre il programma è in esecuzione. Se rilevi un difetto durante l'uso del tuo programma puoi, dal "RunShell", caricare il debugger e controllare lo stato delle variabili in quel momento.

Ora vediamo l'aspetto più importante del nostro sistema di sviluppo: il compilatore. Finalmente un compilatore realmente object-oriented, un vero compilatore C++, non più un semplice compilatore ANSI C adattato alla traduzione di sorgenti C++ in ANSI C.

Infatti questa carenza tra i sistemi di sviluppo Amiga è stata quella che maggiormente ci ha motivati alla realizzazione di un ambiente in grado di compilare più efficientemente i programmi C++.
Per le specifiche stesse del C++ questo nostro sistema di sviluppo è totalmente compatibile con il linguaggio C (ANSI) stesso, risultando un compilatore C comodo e veloce. Ogni sviluppatore sarà in grado di passare alla programmazione object-oriented in C++ in ogni momento, completamente o parzialmente.
Chi già conosce il linguaggio C++ potrà contare su un compilatore conforme alla definizione di Bjarne Stroustrup e allo standard esteso AT&T 3.0.

Il compilatore genera codice per tutti i processori 68000, incluso il 68060. Puoi ottenere un incremento sostanziale di velocità nella compilazione con la pre-compilazione dei file header. Il linker integrato è compatibile con tutti i formati di libreria disponibili (SAS/C, Maxon C++, Amiga Objects,...) ed è tra i più veloci esistenti.

StormC è adatto per tutti i progetti, siano essi gestionali, grafici, musicali o giochi. Per tutti questi StormC diventerà il tuo sistema preferito. La versione dimostrativa ti convincerà a scegliere lo StormC per il futuro; infatti siamo certi che vorrai provare a compilare i tuoi programmi con lo StormC e confrontarli con quelli compilati con il compilatore che usi abitualmente.

Limitazioni della versione "demo":

- Il compilatore compila sorgenti di tutte le dimensioni, però genera file oggetto di lunghezza massima 10 KByte. Il linker genera eseguibili lunghi al massimo 20 KByte.

- I singoli programmi di StormC NON possono essere lanciati da CLI, devono essere usati tramite l'ambiente integrato.

Nel caso che tu trovi una funzione non implementata o qualche problema
di funzionamento, ti chiediamo cortesemente di contattarci direttamente.

Il nostro indirizzo è:

HAAGE & PARTNER COMPUTER GmbH
PO Box 80
61191 Rosbach
Germany

Telefono : 0049 - 6007 - 93 00 50
Fax : 0049 - 6007 - 75 43

Compuserve: 100654,3133
Internet: 100654.3133@compuserve.com
WWW: http://home.pages.de/-haage
WWW: http://ourworld.compuserve.com/homepages/-haage_partner

Puoi anche rivolgerti al nostro rivenditore italiano:

C.A.T.M.U. snc
Casella postale 63
10023 Chieri (TO)

Telefono: 011-9415237 dal lunedì al venerdì dalle 9.30 alle 12.00
            e dalle 14.30 alle 17.00
FAX : 011-9415237 (24h/24h)

Internet: solotre@mbox.vol.it

1.6 StormC.guide/ST_Esigenze

Configurazione minima richiesta

Qui di seguito trovi la configurazione minima per utilizzare lo StormC:

- Amiga con hard disk
- Kickstart e Workbench 2.0 (v37)
- 3 MB RAM
- 5 MB di spazio libero sul tuo hard disk

Con questo computer tu potrai iniziare a programmare con StormC,
ma le dimensioni dei progetti sono limitate. Inoltre non tutte
le caratteristiche del debugger saranno utilizzabili e l'editor
non colora automaticamente le parole chiave.

Invece con la seguente configurazione minima potrai usufruire di
tutte le caratteristiche prodotto e del suo ambiente integrato:

- Amiga con 68030 inclusa MMU
- Kickstart e Workbench 3.0 (preferibile 3.1)
- 8 MB RAM
- 30 MB di spazio libero sul tuo hard disk

Inoltre ricorda sempre la regola: DI PIÙ È MEGLIO!

1.7 StormC.guide/ST_Installazione

Installazione

È usato per l'installazione l'"Installer" AT/Commodore, con il quale
puoi installare sul tuo HD questa versione.
Certamente gli utenti (e sviluppatori) Amiga più esperti non troveranno
difficoltà durante questa fase.

Inserisci nel disk-drive il primo disco della versione "demo" ed esegui
un doppio-click sull'icona del disco stesso.

Prima di iniziare l'installazione è consigliabile che tu legga il file
"Leggimi" che è presente in questo disco. In questo file troverai
importanti integrazioni e suggerimenti che non sono stati inseriti nel
manuale per vari motivi.

Dopo aver letto il file "Leggimi", seleziona con un doppio-click l'icona
"Installa StormC" e attendi che il programma "Installer" e lo script
di installazione siano caricati.
Ora segui le istruzioni del programma di installazione. Se alcuni
passi non ti fossero chiari ti ricordo che puoi selezionare il bottone
"Aiuto..." e riceverai maggiori informazioni.

Se l'installazione avrà successo, riceverai l'appropriato messaggio.

Se l'installazione non sarà ultimata per qualche motivo ripeti l'operazione
chiedendo al programma "Installer" di riportare le azioni eseguite sul
File di sequenza (opzione disponibile nella seconda pagina iniziale del
programma di installazione. Esamina tu stesso questo file per capire
cosa ha ostacolato la corretta installazione in modo da rimediare tu
stesso al problema.
Consultaci telefonicamente nel caso tu non sia riuscito ad eseguire
questa operazione.

1.8 StormC.guide/ST_Problemi

Cosa fare nel caso di problemi "insolubili" con lo StormC

Nel caso che tu abbia riscontrato difficoltà durante l'uso e la procedura di
installazione dello StormC, contattaci direttamente.

 Puoi raggiungerci in vari modi:

HAAGE & PARTNER COMPUTER GmbH

PO Box 80
61191 Rosbach
Germany

Telefono: 0049 - 6007 - 93 00 50 (giorni feriali dalle 9:00 alle 17:00)

Telefax : 0049 - 6007 - 75 43 (24h/24h)

Compuserve: 100654,3133

Internet: 100654.3133@compuserve.com

WWW: http://ourworld.compuserve.com/homepages/haage_partner

Nota che ci devi scrivere in inglese o tedesco.

Oppure puoi contattare il nostro rivenditore italiano:

C.A.T.M.U. snc
Casella postale 63
10023 Chieri (TO)
Italia

Telefono: 011-9415237 dal lunedì al venerdì dalle 9.30 alle 12.00
e dalle 14.30 alle 17.00
FAX : 011-9415237 (24h/24h)

Internet: solotre@mbox.vol.it

con cui puoi dialogare in italiano.

1.9 StormC.guide/ST_Corso

Il corso

Nella seguente sezione del manuale imparerai ogni cosa sull'uso dello StormC. Le caratteristiche essenziali del compilatore ti saranno mostrate attraverso semplici esempi, inizierai così a conoscere le operazioni basilari della gestione dei progetti, come modificare e creare i tuoi file sorgente e come avviare la compilazione. Infine, un esempio passo-passo ti mostrerà come funziona il debugger.

Con questo corso ti mostreremo le potenzialità del sistema di sviluppo dello StormC e potrai valutare se preferirlo agli altri.

1.10 StormC.guide/ST_Iniziamo

Il tuo primo programma con StormC

Ora vedrai come iniziare un nuovo progetto, come modificare il programma sorgente e come compilarlo.

Per prima cosa, esegui lo StormC da Workbench con un doppio-click
sulla sua icona. Trovi questo programma nel cassetto in cui lo hai
installato sul tuo hard disk.

Durante questa fase vedrai un messaggio di benvenuto che rimarrà visibile
mentre ogni componente dello StormC si prepara. Dopo questo messaggio,
completata l'operazione di caricamento, vedrai visualizzata in alto a
sinistra un pannello di controllo rappresentante le principali funzioni
offerte dallo StormC, qui a portata di mano per un facile accesso.

Le icone nel pannello di controllo forniscono le seguenti funzioni:

Nuovo testo
Carica testo
Salva testo

Nuovo progetto
Carica progetto
Salva progetto

Compila
Esegui programma
Avvia il debugger

Inizieremo - naturalmente - con il solito: "Hello world!".

Sorgente-Hello World

1.11 StormC.guide/ST_Progetto

Creazione di un nuovo progetto

Seleziona l'icona "Nuovo progetto". Una nuova finestra di progetto
è stata creata.

Che cos'è un progetto?

Un progetto è l'insieme di tutte le parti relative al tuo programma,
per esempio i file C, C++ e assembler, i file header, i file oggetto, le
librerie, la documentazione, le sue parti grafiche, le immagini e le
sue risorse.
Oltre che fornirti una visione di insieme di tutte queste parti, il gestore
di progetti dello StormC è anche un MAKE con interfaccia utente grafica.

1.12 StormC.guide/ST_Make

Make e la dipendenza tra i moduli

Ad ogni passaggio del compilatore il gestore dei progetti controlla le relazioni tra i vari file in base alle loro estensioni ".o", ".h", ".ass", ".asm", ".i", ".c" e naturalmente anche tra ".cc" and ".cpp".

Così il gestore dei progetti sa da sè che un certo file sorgente C deve essere ricompilato a seguito di una modifica ad un file header che il sorgente C ha incluso.

Ad ogni selezione delle icone "Compila" o "Esegui" tutte le relazioni tra i file sono esaminate. La procedura del Make decide quale modulo del programma deve essere ricompilato.
"Esegui" differisce da "Compila" in quanto dopo la compilazione il programma prodotto è eseguito automaticamente.

Salvataggio del progetto e creazione di una nuova directory

Ora tu dovresti salvare il tuo progetto in una nuova directory. Seleziona l'icona "Salva progetto", quindi la directory "StormC:" ed inserisci come percorso e nome del file "Hello World/Hello World.c". Il suffisso ".¶" sarà inserito automaticamente ed indica che questo file è un progetto realizzato con lo StormC (ricorda che il carattere "¶" è il simbolo di paragrafo e puoi ottenerlo da tastiera con la pressione simultanea di <ALT> + <P>).

La domanda spontanea che ti sarà venuta in mente è: perché salvare un progetto che ancora è privo di ogni contenuto?
Ecco la nostra risposta: perché ti permette di inserire i nomi dei file sorgente in modo relativo al percorso del progetto; in questo modo ti sarà più facile maneggiare l'intero progetto ed il percorso del progetto sarà quello scelto automaticamente nel file requester.

1.13 StormC.guide/ST_Sorgente

Creazione e modifica di file sorgente

Ora possiamo iniziare il nostro progetto.
Apri una nuova finestra "editor" selezionando l'icona "Nuovo testo".

Prima di iniziare a scrivere il programma, ti spieghiamo le parti principali dell'editor che sono in alto nel suo "pannello di controllo".

F = Mostra o nasconde i fine-linea (¶)
T = Mostra o nasconde le tabulazioni (»)
S = Mostra o nasconde gli spazi (·)
I = Attiva o disattiva l'indentazione automatica
V = Attiva o disattiva il modo di sovrascrittura
L = Attiva o disattiva il modo a sola lettura

Il prossimo campo indica se il testo è stato modificato.

A destra del pannello di controllo sono visualizzati la posizione del cursore in righe e colonne ed il numero di linee totali.

Ora puoi scrivere il seguente testo:

```
/ * Hello World
    demo of the preview tutorial * /

#include <stdio.h>

void main(void)
{
   printf("Hello, World!\n");
}
```

Salva questo file con il nome "Hello World.c" nella directory "Hello World".

Seleziona dal menu la voce "Progetto / Aggiungi finestra" e
vedrai che il nome del file comparirà nella sezione sorgente del gestore di
progetto. Questo programma di gestione progetti controlla l'estensione dei
file e li distingue nei vari tipi.

Prima di compilare questo semplice programma dovresti indicare il
nome del programma, altrimenti il suo nome sarà "a.out".

Seleziona nel menu "Progetto" la voce "Scegli nome programma...".
Assicurati che per questa operazione la finestra del gestore di
progetto sia quella attiva.

1.14 StormC.guide / ST_Compila

La compilazione

Seleziona l'icona "Compila". Si aprirà la finestra relativa agli errori di
compilazione e la traduzione del sorgente avrà inizio. Durante la
compilazione alcuni messaggi potrebbero essere visualizzati nella
finestra di errore.

Se si verificasse un errore, una descrizione sarà fornita in questa
finestra con l'indicazione della possibile causa dell'errore ed il
numero di linea del sorgente in cui si è verificato l'errore.

La semplice selezione con doppio-click sul messaggio d'errore ti
porterà nella finestra dell'editor alla linea presunta errata.
Esegui le tue correzioni e compila di nuovo.

1.15 StormC.guide / ST_PRGStart

Esecuzione del programma

Dopo una compilazione corretta e la relativa operazione di link del tuo
programma, il tasto "Esegui" diventa accessibile. Selezionalo o usa
l'icona "Esegui" dal pannello di controllo.

Se hai selezionato "Esegui" anziché "Compila" il gestore di progetto, dopo
aver appurato che il programma è stato compilato, esegue egli stesso
il programma.

RunShell

Avviando il programma dal gestore di progetto c'è qualcosa di più caratteristico che ora ti mostro:

Avrai notato che quando il tuo programma inizia l'esecuzione, un'altra finestra si apre. Questa nuova finestra si chiama "RunShell".

Naturalmente potrai eseguire semplicemente il tuo programma, ma durante la fase di sviluppo spesso ti servirà un controllo completo sul programma in esecuzione. Con il RunShell dello StormC è possibile tracciare e controllare il programma durante la sua esecuzione. Se si verifica un errore che genera una eccezione a livello CPU, il RunShell (nella maggior parte dei casi) prende il controllo e ti permette di esaminare l'errore ed il sorgente che ha generato il problema.

Inoltre, caratteristica unica del RunShell è la sua abilità di tracciamento delle risorse. Per fare questo, il RunShell memorizza tutte le chiamate a molte funzioni in grado di prendere possesso di risorse di sistema (come AllocMem(), OpenWindow(), Open(), ...). Quando il programma termina la propria esecuzione, il RunShell controlla se tutte le risorse sono state rilasciate e se lo sono state più di una volta. Puoi quindi esaminare direttamente il sorgente per apportare le modifiche necessarie.

Inoltre il RunShell offre la possibilità di inviare i segnali Ctrl-C, Ctrl-D, Ctrl-E, Ctrl-F, che di solito devono essere inviati da Shell.

Puoi anche modificare la priorità del programma con ogni valore tra -128 e +127, mettere in pausa il programma tramite il bottone di "Pausa", da notare che questo bottone agisce come un interruttore: riselezionandolo il programma continua la sua esecuzione.

Con la selezione del bottone "Termina", il programma esaminato viene terminato e tutte le sue risorse sono restituite al sistema (segnali e memoria rilasciate al sistema, schermi e finestre, requester e file sono chiusi), con il piacevole risultato di non avere il sistema instabile per la terminazione imprevista di un programma che hai verificato bloccato in un ciclo senza uscita o in attesa di segnali che mai gli sarebbero arrivati, o nei guai per altri problemi.
Queste sono condizioni che si verificano frequentemente durante lo sviluppo, e questa caratteristica del RunShell ti risparmierà certo molto tempo, evitandoti di resettare l'Amiga, di ricrearti assegnamenti o altri settaggi che magari sono indispensabili al funzionamento del tuo programma.

Nel nostro esempio "Hello World", il programma è semplice ed è eseguito così rapidamente che questa finestra RunShell è immediatamente richiusa. Il prossimo esempio ti mostrerà meglio come RunShell funziona con il debugger.

La finestra di output ("Hello World") che è ancora aperta sul tuo schermo è una normale console come quelle della shell. È stata aperta dal programma quando ha usato la funzione "printf". Per chiuderla semplicemente usa il solito gadget di chiusura in alto a sinistra.

1.16 StormC.guide/ST_Debug

Il debugger

Il debugger è una necessità per trovare velocemente dove il programma
è errato, come genera l'errore e quando.
Il debugger ti consente di definire breakpoint nei tuoi programmi,
osservare come cambiano i valori assunti dalle sue variabili, strutture e
classi. In questo modo gli errori possono essere circoscritti, individuati
e rimossi con rapidità.

Per dimostrarti le operazioni del debugger, per semplicità, devi caricare
un progetto che abbiamo preparato noi per te e compilarlo.

Nella directory "Esempi" (examples) troverai il file "Colorwheel".
Aprilo (con doppio-click sulla sua icona dal Workbench): il progetto
sarà caricato e visualizzato.
Seleziona nel menu "Impostazioni" la voce "Progetto".
Premi sul gadget ciclico fino a quando apparirà "Opzioni C/C++".
Come vedi, "Debug completo" è attivato.

Seleziona l'icona debug. Il sistema controllerà se tutti i moduli sono
compilati con l'opzione "debug" o se devono essere ricompilati.
Vedrai che l'icona debug esegue la stessa procedura come l'icona "Esegui",
Dopo la fase di link, il programma sarà eseguito in modalità "debug".

In funzione delle opzioni scelte, si apriranno le finestre dei moduli,
delle variabili correnti, delle variabili ispezionate.

Inoltre il modulo che contiene la funzione "main()" è aperto e il suo
sorgente visualizzato sulla posizione di inizio della funzione main()
stessa.

Noterai che il contenuto della finestra editor è cambiato un po'.
La prima colonna del testo è spostata un po' a destra lasciando una colonna
intera vuota che può essere usata per mostrare i breakpoint.
I breakpoint indicano le posizioni in cui si richiede che si fermi
l'esecuzione del programma. È sufficiente puntare su una posizione
in questa colonna per determinare l'inserimento di un indicatore "X"
che evidenzia l'attivazione di un breakpoint in quella posizione.

Inserisci un breakpoint subito dopo una chiamata alla funzione "OpenScreen".
Se la finestra "variabili correnti" non è aperta, aprila tu selezionando
dal menu "Finestre" la voce "Variabili...".

Seleziona l'icona "Vai avanti fino al prossimo breakpoint" sul pannello
di controllo di RunShell.

Da quando hai inserito il breakpoint ad ora, il programma ha eseguito una
chiamata ad "OpenScreen()", ed ora ha interrotto la sua esecuzione.
Un nuovo schermo è stato aperto ed il programma si è fermato.

La prossima funzione "GetRGB32()" fornisce un array di unsigned long
con i dati colore. Esaminiamo meglio questo array.

Innanzi tutto dobbiamo inserire questa variabile nella finestra di ispezione.

Dalla finestra "Variabili" seleziona la variabile "colortable"
con il mouse ed esegui un click sul simbolo "Ispeziona" nella parte alta della
finestra.

Esegui tre passi del programma (nel modo a passo singolo), ed osserva
come questa variabile "colortable" cambia nella finestra di ispezione. Nota
quanto è semplice analizzare le variabili con questa finestra di ispezione.

Vorrai ora inserire più breakpoint e divertirti con il debugger e le
sue funzioni. Sono certo che tutto questo ti sarà familiare in
breve tempo.

Per uscire dal debugger, termina il programma in esecuzione in modalità
debugger. Tutte le sue finestre saranno chiuse automaticamente e così
anche RunShell.

1.17 StormC.guide / ST_Sezioni

Sezioni di un progetto

I file in un progetto sono catalogati automaticamente in sezioni a seconda
della estensione del loro nome, e nel caso di documentazione dal nome
intero. Se aggiungi un file ".c" ad un nuovo progetto, il gestore di
progetto crea una nuova sezione chiamata sorgente contenente questo file.
Quando aggiungi altri sorgenti, questa sezione viene semplicemente espansa.

Le sezioni predefinite e riconosciute sono le seguenti:

Sorgenti
  ".c"
  ".cc"
  ".ccp"
  ".c++"
  ".cpp"

Header
  ".h"
  ".hh"
  ".hhp"
  ".h++"
  ".hpp"

Sorgenti ASM
  ".asm"
  ".ate"
  ".s"

Header ASM
  ".i"

Documentazione
  ".dok"
  ".doc"
  ".txt"
  ".readme"

```
  "read me"
  "liesmich"
  "readme"
  "read me"
  "read.me"
  "lies.mich"
  "leggimi"

ARexx
  ".rexx"

Altri
  "*"

Progetti
  ".¶"

AmigaGuide
  ".guide"

Eseguibile
```

1.18 StormC.guide / ST_Caratteristiche

```
Caratteristiche dello StormC

Nonostante l'esistenza delle specifiche standard ANSI C, ciascun
compilatore ha le proprie peculiarità.  Queste sono precedute
dalla direttiva "#pragma". Come le direttive "#include" anche le
"#pragma" sono interpretate ed eseguite dal preprocessore.


I modi di compilazione

#pragma -

è una estensione non standard che impone al compilatore di attenersi
strettamente alle specifiche ANSI-C.

#pragma +

è una estensione che permette al compilatore di accettare
il successivo sorgente in C++.

RAM Chip e Fast

L'architettura Amiga è poco ortodossa per alcuni aspetti; per esempio ci
sono diverse classi di RAM.

Normalmente il programmatore si chiede come allocare dati che devono
risiedere nella memoria Chip. Questo è indispensabile per immagini
e suoni.
Lo StormC offre come soluzione due pragma "chip" e "fast".

#pragma chip
```

tutti i dati dichiarati dopo questa direttiva sono allocati nella Chip RAM.

#pragma fast

i dati allocati dopo questa direttiva sono allocati in ogni tipo di memoria, preferibilmente, se disponibile, nella memoria Fast.

Codice interrompibile

Con lo StormC è possibile l'inserimento automatico di codice di controllo in ogni ciclo (almeno una volta per ciclo) al fine di permettere l'uscita anche da cicli senza fine (normalmente non desiderati dall'utente, nè dallo stesso programmatore).

Con questa caratteristica il programma controlla se gli è stato inviato un <CTRL> + <C> o dall'utente o dal CLI tramite il comando "break".

È inutile dire che questa opzione rallenta molto l'esecuzione del programma, ma in alcuni casi, soprattutto nelle fasi di test dei programmi, può evitare molti problemi.

Se vuoi dichiarare interrompibile anche solo una sezione di codice, puoi usare la direttiva

#pragma break +

e disattivare questa funzione nel sorgente dove ritieni più opportuno con

pragma break -

Se premi <CTRL> + <C> mentre è in esecuzione una parte dichiarata interrompibile, il programma termina come se fosse eseguita una chiamata alla funzione "exit(900)".
Tutti i file sono chiusi, la memoria e le altre risorse restituite al sistema e le funzioni passate ad atexit() sono invocate.
Attenzione, come per ogni normale "exit()", nessun distruttore è chiamato.

Le chiamate al sistema operativo

Le funzioni al sistema operativo Amiga sono chiamate tramite #pragma amicall.

Tali dichiarazioni sono così costruite in quattro parti:

- nome della variabile puntatore alla base della libreria.
- offset della funzione con intero positivo.
- nome della funzione (deve essere già stata dichiarata); per evitare ambiguità questi nomi non possono essere overloaded.
- lista dei parametri (rapresentati da una corrispondente lista di nomi di registri tra parentesi)

Per esempio:

```
#pragma amicall(Sysbase, 0x11a, AddTask (a1,a2,a3))
#pragma amicall(Sysbase, 0x120, RemTask (a1))
#pragma amicall(Sysbase, 0x126, FindTask (a1))
```

Normalmente tu non scriverai mai queste dichiarazioni perché sono incluse nei file header Amiga.

Unione di linee

Sia per il linguaggio C che per il C++ le linee non sono di per sé molto significative, tuttavia per il preprocessore le linee sono molto significative: ogni singola direttiva può essere lunga solo una linea. Per ovviare a questo, viene usato il carattere "\" che unisce virtualmente la linea alla successiva. Esempio:

```
# define QUALCOSA \
47081115
```

Questa è una valida definizione di costante, e il numero "47081115" è forzato come appartente alla linea precedente.

Simboli predefiniti

Il preprocessore conosce molte macro predefinite.
Alcune sono definite dallo standard C ANSI, altre sono parte del C++, altre ancora sono tipiche dello StormC.
Queste macro non possono essere ridefinite.

__COMPMODE__

è definito in StormC come "0" se il compilatore è in modalità C
è definito in StormC come "1" se il compilatore è in modalità C++

__cplusplus

Con lo StormC la macro "__STDC__" è definita sia in modalità C che in modalità C++. Se vuoi controllare se il tuo sorgente è compilato in modo C++, lo puoi fare con questa macro "__cplusplus".

__DATE__

È espansa con la data di compilazione. Questo è molto utile se vuoi fornire un programma con un unico numero di versione:

```
#include <stream.h>
void main ()
{  cout  « "version 1.1 from " __DATE__",  "__TIME__" clock\n;  }
```

La data è fornita nella forma mese - giorno - anno ,
per esempio "Feb 08 1996"; l'ora è fornita nella forma "hh:mm:ss".

__FILE__

Questa macro contiene il nome del file sorgente ed è una variabile

```
stringa.

Esempio:

# include <stream.h>
void main ()
{  cout «  "This is line " « __LINE__ « " in the file " __FILE__ ".\n";  }

Il valore della macro __FILE__ è una costante stringa di caratteri e
può essere unita ad altre stringhe.

 __LINE__

La macro __LINE__ restituisce il numero di linea nella quale è usata,
tale numero è una costante numerica intera.

 __STDC__

Questa macro restituisce il valore numerico "1" se il compilatore è
ANSI C.  Altrimenti non è definita.

 __STORM__

Questa macro restituisce il nome del compilatore e il numero di versione.

 __TIME__

(see __DATE__)
```

1.19 StormC.guide / ST_MenuComandi

```
Comandi di Menu

Progetto

Nuovo
Quando viene selezionato dall'icona o dalla finestra progetto, un nuovo
progetto è aperto; questo corrisponde ad effettuare un click sull'icona
"nuovo progetto".

Se la finestra attiva è una finestra editor, una nuova finestra editor
è aperta; la stessa cosa si verifica effettuando un click sull'icona
"nuovo testo"

Apri...
Se l'icona o la finestra del progetto è attiva, un progetto verrà
aperto. Quando viene selezionato dalla finestra editor un nuovo file di
testo, questo viene caricato. Altrimenti, un file requester standard
ASL comparirà, chiedendo un file di input. Puoi anche scegliere Apri...
dal pannello di controllo.

Salva
Se la finestra del progetto è attiva, il progetto viene salvato. Questo
corrisponde ad effettuare un click sull'icona "Salva Progetto".
```

Se una finestra editor è attiva, il suo testo viene salvato. Questo corrisponde ad effettuare un click sulla icona "Salva Testo".

Salva come...
Il file requester Salva è aperto; qui puoi selezionare il nome di un file per il tuo progetto o il tuo testo. A seconda che la finestra attiva sia quella dell'editor o quella del progetto, puoi salvare o il progetto o il sorgente. Il pannello di controllo ti offre due icone: una per salvare il progetto e l'altra per il testo.

Salva come modello di progetto
Questa voce è solo accessibile dalla finestra progetto. Consente di salvare delle impostazioni che saranno utilizzate in futuro per i prossimi progetti.

Puoi predefinire tutte le impostazioni di progetto (relative ad ambiente C/C++, pre-processore C/C++, opzioni C/C++, messaggi di errore C/C++, opzioni del linker e di avvio del programma) e naturalmente tutte le sezioni del gestore di progetto.

Salva tutto
Con questa voce del menu salvi tutti i sorgenti ed i progetti che non sono ancora stati salvati su disco. Se qualcosa è ancora privo di nome, comparirà un file requester per ogni nome mancante.

Aggiungi file...
Questa voce di menu è accessibile solo dalla finestra di progetto. Un file può essere selezionato dal file requester e sarà importato nel gestore di progetto nella sezione adeguata in funzione del estensione del suo nome.

Aggiungi finestra
Questa voce di menu è accessibile quando c'è un progetto e la finestra editor è attiva. Con un semplice click il file della finestra editor attiva è aggiunto al progetto nella opportuna sezione. A differenza della voce "Aggiungi file" il file requester non apparirà.

Scegli il nome programma...
Naturalmente ad ogni programma serve un nome. Poiché un programma può essere composto da diversi moduli, la scelta del nome non può essere automatica. Con l'aiuto del file requester puoi inserire il nome del programma e scegliere una giusta collocazione sul tuo disco.

Chiudi
A seconda che la finestra attiva sia quella dell'editor o quella del progetto, l'una o l'altra sarà chiusa. Ti comparirà un file requester se non è stato salvato il contenuto della finestra da chiudere. La stessa cosa succede se usi il gadget di chiusura della finestra.

Informazioni
Ti comparirà un requester con le informazioni sul prodotto, il copyright, il numero di telefono e l'indirizzo Internet del produttore e dell'importatore.

Esci
Esci dallo StormC. Se qualcosa non è stato salvato, comparirà un messaggio di avviso.

Composizione

Definisci blocco
Questa voce di menu è disponibile solo dalle finestre editor.
Attiva la modalità di definizione blocco dell'editor.

Taglia
Questa voce di menu è disponibile solo dalle finestre editor.
L'area di testo selezionata con la modalità "definisci blocco" è
rimossa dal testo e copiata negli appunti.

Copia
Questa voce di menu è disponibile solo dalle finestre editor.
Come "taglia", copia il blocco di testo selezionato negli appunti però
senza rimuoverlo dal testo.

Incolla
Questa voce di menu è disponibile solo dalle finestre editor.
Inserisce il testo contenuto negli appunti all'interno del testo alla
posizione corrente del cursore.

Cancella
Questa voce quando è scelta da una finestra dell'editor, rimuove la
parte di testo selezionata con "definisci blocco", ma a differenza di
Taglia, non inserisce quella parte di testo negli appunti. Puoi ancora
recuperare il testo attraverso la funzione "Annulla".

Quando questa voce del menu è scelta dalla finestra progetto, il modulo
indicato è rimosso dalla sezione del progetto senza possibilità di
annullare l'operazione tramite la voce "Annulla".

Annulla
Questa voce del menu è disponibile solo dalle finestre dell'editor.
Essa rimuove gli effetti dell'ultimo comando impartito all'editor.

Rifai
Questa voce del menu è disponibile solo dalle finestre dell'editor.
Con "Rifai" tu puoi rieseguire l'operazione appena annullata con "Annulla".

Trova e sostituisci...
Questa voce del menu è disponibile solo dalle finestre dell'editor,
ovviamente. Ti apparirà un requester dal quale potrai indicare quale
testo cercare e quale testo usare per rimpiazzarlo.

Questa opzione può essere utilizzata anche solo per la ricerca: basta
omettere il testo da sostituire ed impartire il comando con il gadget
"Trova". Puoi anche indicare in quale direzione effetturare la ricerca.

Usando il gadget ciclico puoi impostare altri dettagli per la ricerca,
ad esempio ignorare accenti e maiuscole.

Con i tre gadget sul bordo inferiore puoi effettuare la ricerca in
diversi modi.

"Trova" semplicemente cerca la stringa

"Sostituisci" sostituisci la stringa cercata con la nuova stringa voluta.
"Sostituisci tutto" rimpiazza tutte le stringhe trovate con quella voluta.

Trova il prossimo
Questa voce del menu è disponibile solo dalle finestre dell'editor.

"Trova il prossimo" cerca la prossima stringa che coincide con quella cercata l'ultima volta, ma senza aprire il requester.

Sostituisci il prossimo
Questa voce del menu è disponibile solo dalle finestre dell'editor.
"Sostituisci il prossimo" ripete la ricerca con sostituzione come l'ultima volta, senza aprire il requester.

Compila

Compila...
La voce del menu "compila..." è disponibile se esiste un sorgente selezionato nella opportuna sezione del progetto. Puoi anche utilizzare questa voce se il sorgente della finestra editor attiva è presente nel progetto corrente. Con questa funzione puoi anche compilare i singoli moduli.

Make...
Questa voce del menu è disponibile solo dalla finestra progetto o da quella relativa agli errori.
Puoi anche utilizzare questa voce se il sorgente della finestra editor attiva è presente nel progetto corrente. Con questa funzione puoi anche compilare i singoli moduli.

"Make" compila tutti i moduli che sono stati alterati dopo l'ultima compilazione rispettando le relazioni e dipendenze tra programma, sorgenti e file header. Selezionando il gadget "Compila" si ha lo stesso effetto.

Esegui...
Questa voce del menu è disponibile solo dalla finestra progetto o da quella relativa agli errori.
Puoi anche utilizzare questa voce se il sorgente della finestra editor attiva è presente nel progetto corrente.

Se una versione aggiornata del programma già esiste, sarà eseguita. Altrimenti verrà creata come se fosse impartito il comando "Make", dopo il programma sarà eseguito.
Selezionando il gadget "Esegui" si ha lo stesso effetto.

Debug...
Questa voce del menu è disponibile solo dalla finestra progetto o da quella relativa agli errori.
Puoi anche utilizzare questa voce se il sorgente della finestra editor attiva è presente nel progetto corrente.

"Debug..." agisce come "Esegui" e avvia il debugger. Il programma sarà fermato alla prima funzione (main()) pronto per essere esaminato durante l'esecuzione. Selezionando il gadget "Debug" si ha lo stesso effetto.

Imposta data
Questa voce del menu è disponibile solo dalla finestra progetto.
Se usi "Imposta data" su un file sorgente il gestore di progetti
tratterà quel file come se fosse stato modificato. La prossima volta
che il comando "Make" sarà eseguito questo file sarà certamente
ricompilato.

Imposta data per tutti
Questa voce del menu è disponibile solo dalla finestra progetto.
Agisce come "Imposta data", ma su tutti i file sorgente del progetto.
La prossima volta che il comando "Make" sarà eseguito tutti i file
sorgente saranno ricompilati.

Finestre

Errori...
Questa voce del menu è disponibile solo dalla finestra progetto.
Apre la finestra errori.

Moduli...
Questa voce di menu è disponibile solo se il debugger è attivo.
Apre la finestra dei moduli dalla quale è possibile stampare le
informazioni. Questo normalmente include tutti i file della categoria
sorgenti presenti nella finestra progetto per i quali un file debug è
stato creato. Anziché i nomi dei sorgenti troverai i nomi dei moduli
oggetto.

Variabili...
Questa voce di menu è disponibile solo se il debugger è attivo.
Apre la finestra mostrante le variabili attive.

Variabili ispezionate...
Questa voce di menu è disponibile solo se il debugger è attivo.
Apre la finestra mostrante le variabili ispezionate.

Impostazioni

Progetto...
Queste opzioni sono disponibili solo quando il corrispondente progetto
o finestra di errori sono attivati.

Percorso file include e file header precompilati
Il preprocessore è un programma che elabora i file sorgente prima del
compilatore. Il preprocessore prende le definizioni, per esempio quelle
delle funzioni della libreria standard dai file include. Usa la
direttiva "#include" per fare questo. Come tu sai, ci sono due modi per
fare questo:

Se si indicano i nomi dei file tra doppi apici (#include "così"), il
preprocessore cerca i file nella directory corrente. Se, invece,
racchiudi i nomi tra i simboli < > (#include <così>), questi file
saranno cercati tra i file include standard e caricati dalle
appropriate directory. Nel "Path per gli include" puoi scegliere una o
più directory in cui cercare i file include standard.

Un modo per velocizzare la compilazione sensibilmente risparmiando RAM è l'uso di file header pre-compilati. Per usare questa caratteristica, devi indicare al compilatore dove nel tuo sorgente finisce l'intestazione (header) e dove inizia il tuo programma con la direttiva "#pragma header". Semplicemente puoi raggruppare tutti le direttive #include dei file standard in testa al sorgente e scrivere al fondo di questa sezione la direttiva "#pragma header", di seguito le direttive #include dei file propri del tuo programma e quindi il resto del sorgente vero e proprio.

La direttiva #pragma header non avrà nessun effetto se non selezioni l'opzione "Scrivi header" e ricompili gli header modificati. Non appena il compilatore raggiunge la direttiva #pragma sono salvati i files header precompilati nel corrispondente cassetto indicato.

Prima che tu richiami il compilatore la prossima volta, semplicemente attiva "Leggi header". Il compilatore leggerà i file header, cercherà la direttiva "#pragma header" nel sorgente e da qui inizierà la sua traduzione.

Puoi trovare queste opzioni nella finestra di dialogo nel gadget ciclico proprio sotto il bordo superiore della finestra.

Preprocessore

Qui puoi decidere quali warnings devono essere generati dal preprocessore e quali simboli devono essere predefiniti.

I warnings del preprocessore sono facili da configurare, ma per le definizioni dei simboli c'è da fare un po' d'attenzione. Ciascun simbolo che inserisci sarà trattato proprio come le direttive #define che inserisci nei tuoi sorgenti. Questo ti permette di avere alcuni simboli definiti globalmente come ad esempio "DEBUG" o "TRUE" e "FALSE".

Generazione di codice eseguibile e codice di debug

Qui tu puoi selezionare il modo di traduzione del sorgente (ANSI C o C++) e, nel caso di C++, la gestione dei template e l'uso della gestione delle eccezioni.

Il prossimo gadget ciclico ti abilita o meno la generazione del file di debug. Se tu desideri il compilatore genera i file con suffisso ".debug" e ".link" necessari al debugger; essi descrivono le relazioni esistenti tra sorgente e programma eseguibile.

Se desideri lavorare con un debugger simbolico a livello assembler, c'è l'opzione per inserire l'hunk dei simboli nel programma stesso.

Puoi anche produrre sorgente assembler. Il compilatore genera i file ".o" ".s" che contengono il sorgente assembler con il sorgente C relativo nei commenti.

Se abiliti la "generazione di codice interrompibile" il compilatore inserisce dei controlli per il <CTRL> + <C> in ogni ciclo.

L'opzione "ottimizza codice" fa in modo che il programma risulti più compatto e normalmente più veloce.

Il gadget ciclico successivo determina per quale processore il codice deve essere generato. Attenzione che se compili ad esempio per 68060 il programma eseguibile ottenuto non funzionerà su un normale A2000.

Ora puoi scegliere tra generazione di codice per FPU o chiamate alle funzioni matematiche del sistema operativo Amiga.

Ultimo ma non meno importante è il gadget ciclico che ti consente di mutare il modello di memoria utilizzato dal programma generato.

I warning

Lo StormC distingue otto tipi di warning che puoi abilitare o disabilitare a seconda delle tue necessità.

Le impostazioni del linker

Il percorso per la ricerca delle librerie può essere inserito qui; questo è simile al settaggio del percorso per i file header da includere.

Il prossimo gadget ciclico ha tre opzioni:

"Link completo" esegue il link del programma e delle librerie usate.
"No link" non avvia il linker.
"Link senza codice di startup" questo è usato per generazione di librerie condivise o device drivers. I programmi così generati non possono essere eseguiti nè da shell ne da Workbench.

Puoi anche decidere se il linking deve avvenire se si è verificato un errore in fase di compilazione.

Esecuzione

Se tu esegui un programma dall'ambiente integrato StormC puoi specificare quali argomenti sulla linea di comando devono essere passati al programma e quale dimensione dello stack deve essere data al programma.

Carica impostazioni...

Puoi caricare le impostazioni del debugger con il suffisso "RUN". Questa funzione è disponibile solo se il debugger è attivo.

Salva impostazioni

Puoi salvare le impostazioni della attuale configurazione del debugger (quali finestre sono aperte e le loro rispettive posizioni). Il file "StormSettings.Run" è salvato nel cassetto dello StormC stesso. Questa funzione è disponibile solo se il debugger è attivo.

Salva le impostazioni come...

Memorizza le impostazioni della attuale configurazione del debugger come l'opzione sopra, ma puoi opportunamente decidere tu il nome e posizione del file. Questa funzione è disponibile solo se il debugger è attivo.