

Corso di

Misure per l'Automazione e la Produzione Industriale

(Studenti Ingegneria della Produzione Industriale III anno NO)

Il LabVIEW – Lez.3

Marco Laracca
m.laracca@unicas.it



*Gruppo Misure Elettriche ed
Elettroniche*

*Facoltà di Ingegneria, DAEIMI.
Università degli Studi di Cassino*

Corso di Strumentazione Virtuale

LabView

Parte 3

- **Stringhe**
- **I/O su file**
- *Variabili locali e globali*
- *Property nodes*
- *Applicazioni*

Le Stringhe

- Una stringa è una sequenza di caratteri
- Ogni carattere ha una corrispondente codifica numerica in byte (8 bit) secondo lo standard ASCII
- I primi 32 dei 256 caratteri ASCII non sono visualizzabili (controllo)
- Gli ultimi 128 caratteri costituiscono il set *esteso*
- Impieghi: visualizzazione di messaggi, I/O su file, controllo di strumentazione
- Esistono controlli e indicatori di tipo stringa

<i>Stringa:</i>	V	D	C		-	1	2	.	4	5	6
# <i>ASCII:</i>	86	68	67	32	45	49	50	46	52	53	54

Display Modes per gli indicatori stringa

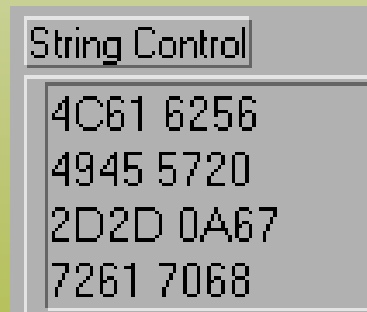
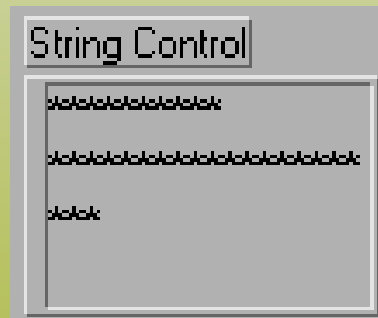
- **Normal display**



- **\ code display**



- **Password display**
- **Hex display**



Backslash codes per alcuni caratteri di controllo:

\b backspace

\s spazio

\r return (CR)


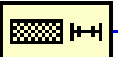

\n new line (LF)

\t tab

Funzioni per le stringhe (1)





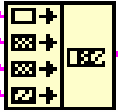

String Length

string  length

String    Length = 20

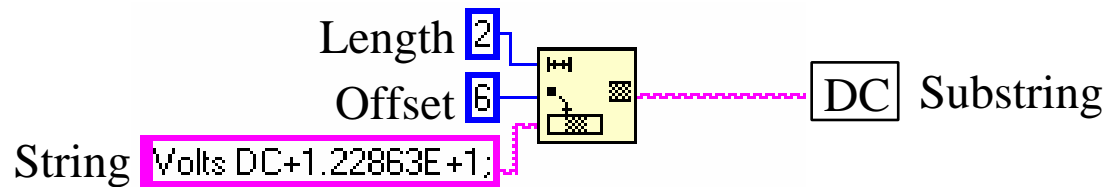
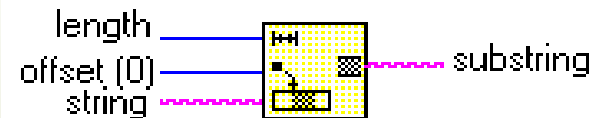
Concatenate Strings

string0  concatenate of string0,string1, ...
string1

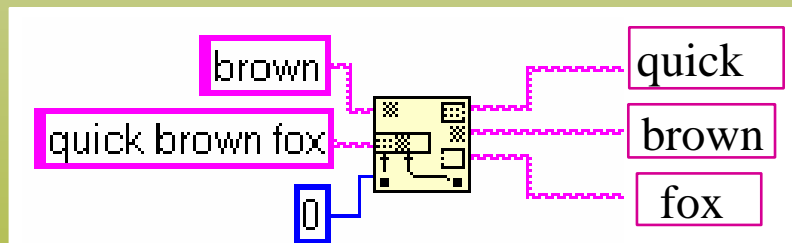
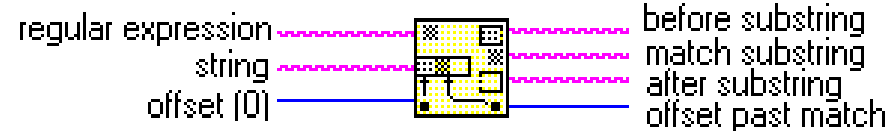
Strings      
Concatenated String

Funzioni per le stringhe (2)

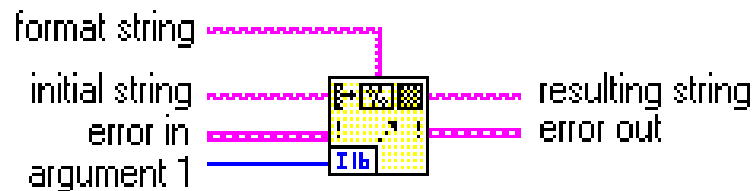
String Subset



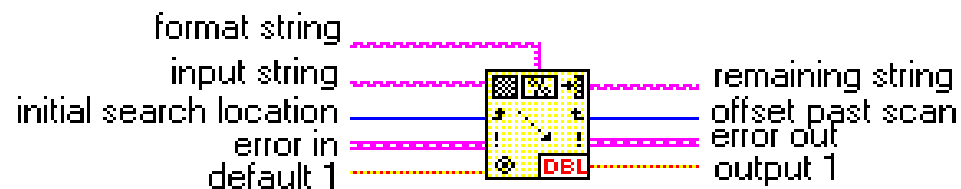
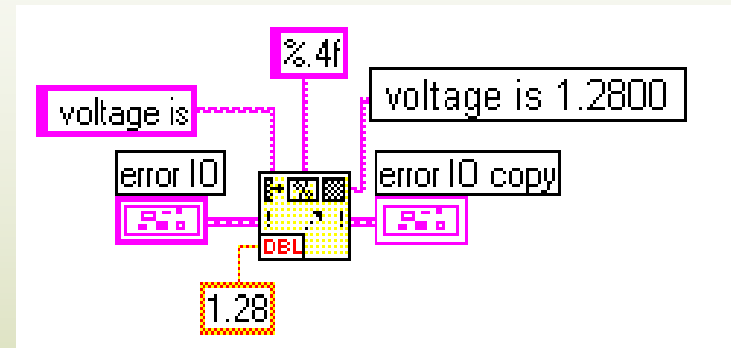
Match Pattern



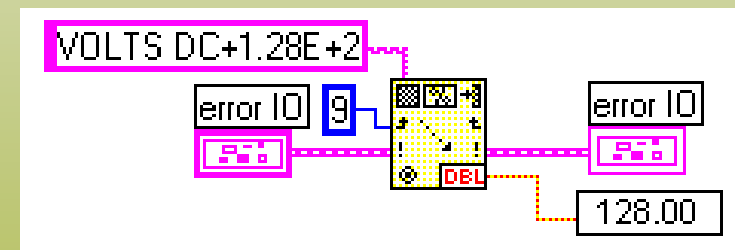
Funzioni per le stringhe (3)



Format Into String
(è resizable)



Scan From String
(è resizable)



Le stringhe di formato

- „ Indicano il formato, il tipo di dato, il numero di caratteri, l'allineamento ed il numero di cifre decimali (per i *floating point*) con cui visualizzare un dato
- „ Si usano come nella funzione **printf()** del linguaggio C
- „ Esempio:

Risultato = %.1f

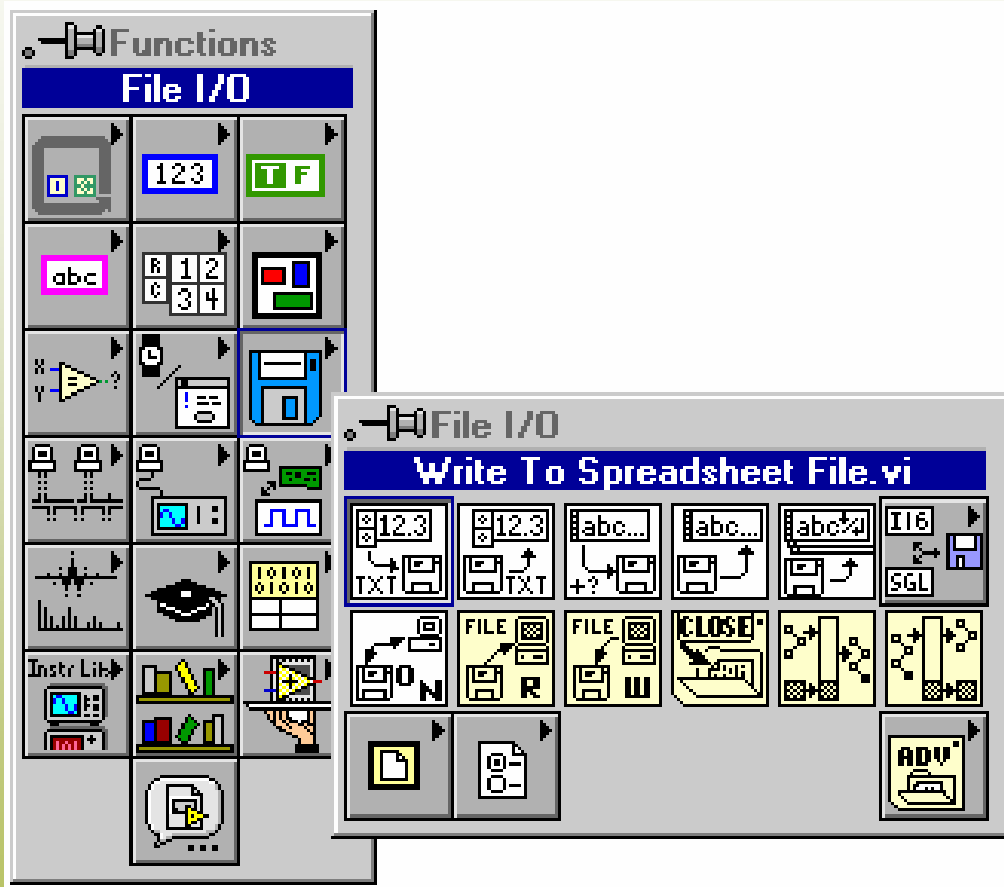


Risultato = 12.5

%d	intero con segno
%u	intero senza segno
%s	stringa
%f	floating point (15.012)
...	...

- „ Dal menu di contesto, “*Edit Format String*” apre una finestra di dialogo che facilita la preparazione della stringa di formato

Funzioni per l' I/O su File



Alto livello:

- Read/Write to spreadsheet file
- Read/Write characters to file
- Read lines from file
- Read/Write to binary file

Livello intermedio:

Open, Read, Write, Close

Livello avanzato:

- Gestione directory
- Dipendenti dal S.O.
- File dialog

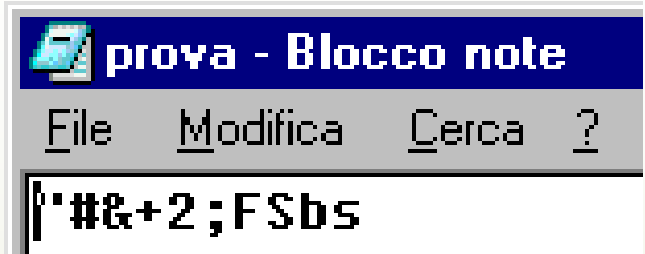
Funzioni per i file di livello intermedio

Gestione diretta delle operazioni sui file:

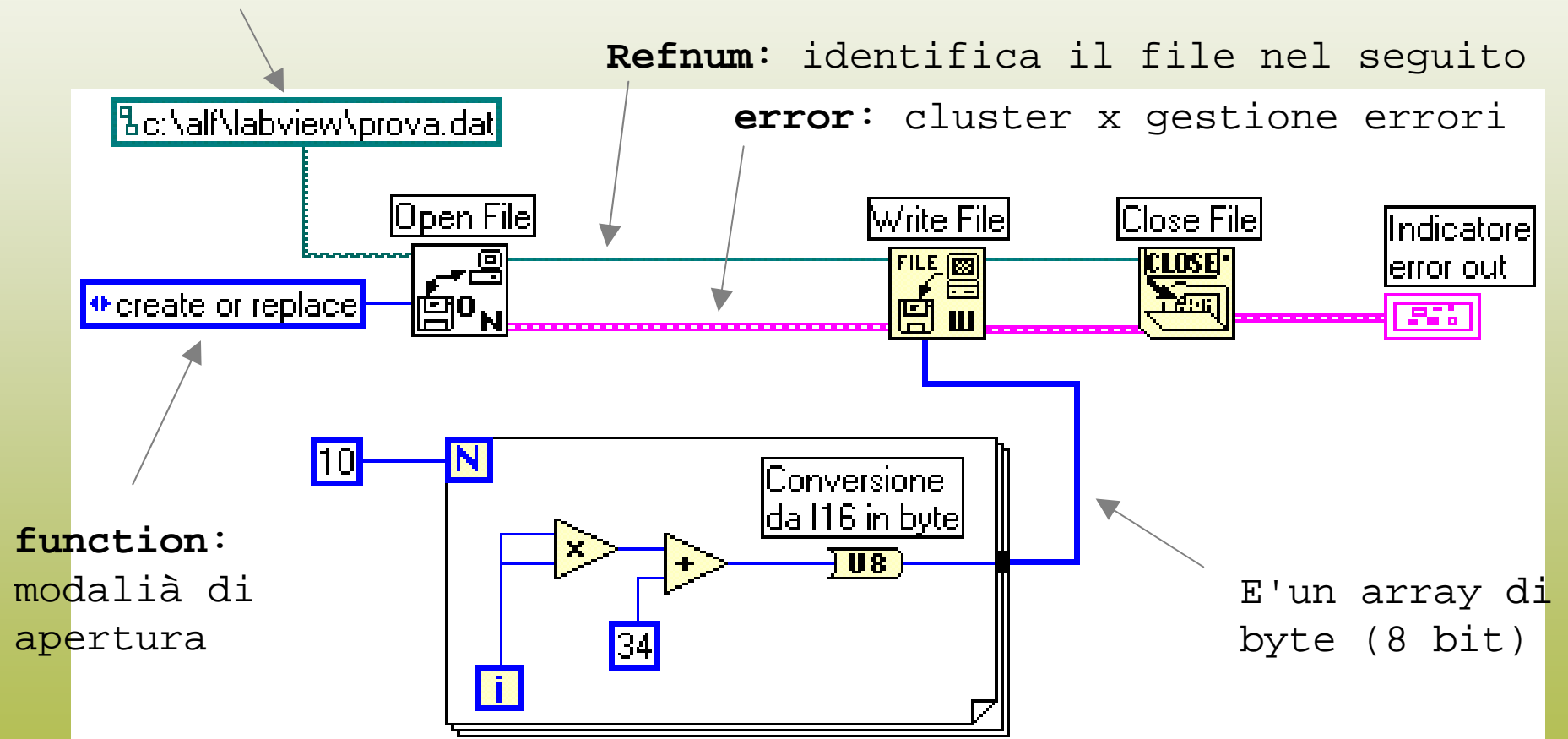
- . Apertura del file: *Open/Create/Replace File*
- . Lettura o scrittura: *Read File e Write File*
- . Chiusura del file: *Close File*
- . Gestione errori

Scrittura su file

Funzioni di livello intermedio



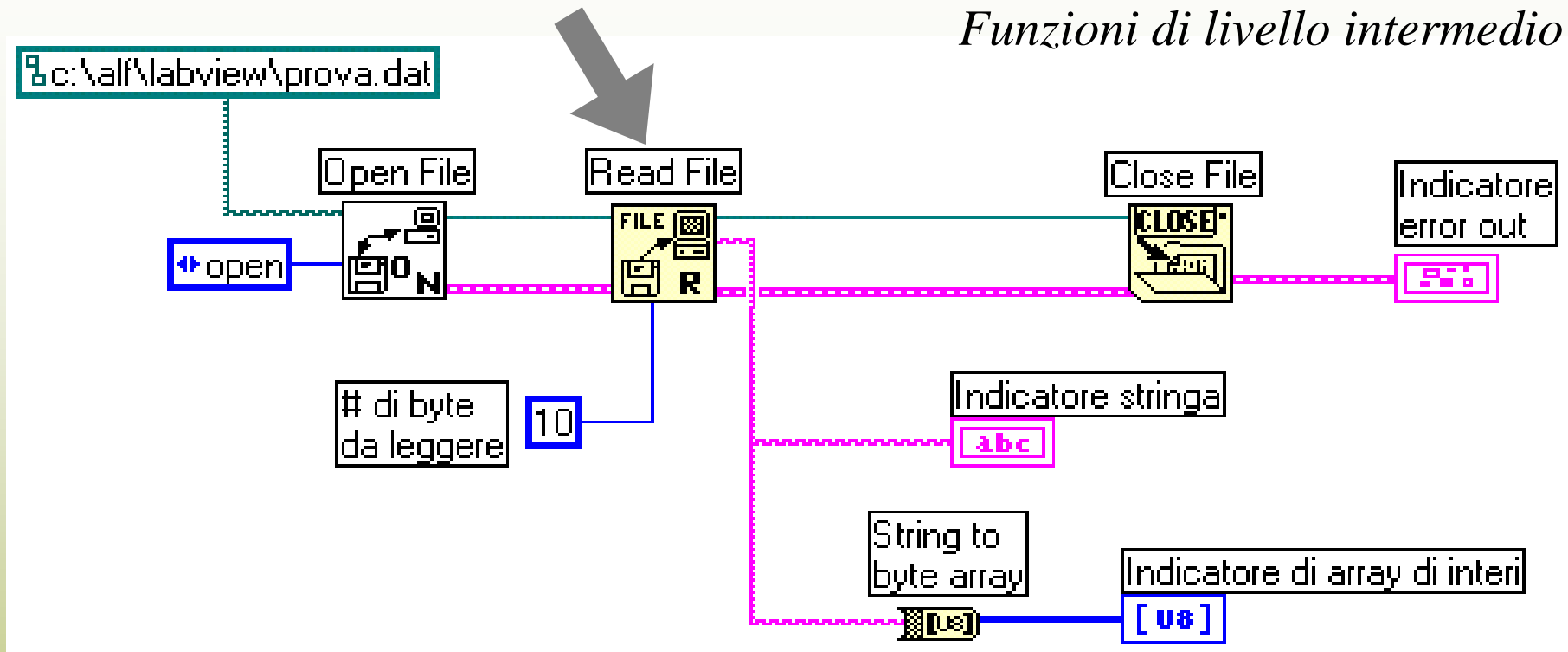
File path: il formato dipende dal s.o.



La *Read File* per default restituisce stringhe

Lettura da file

Funzioni di livello intermedio



Indicatore di array di interi

0	34	35	38	43	50	59	70	83	98	115
---	----	----	----	----	----	----	----	----	----	-----

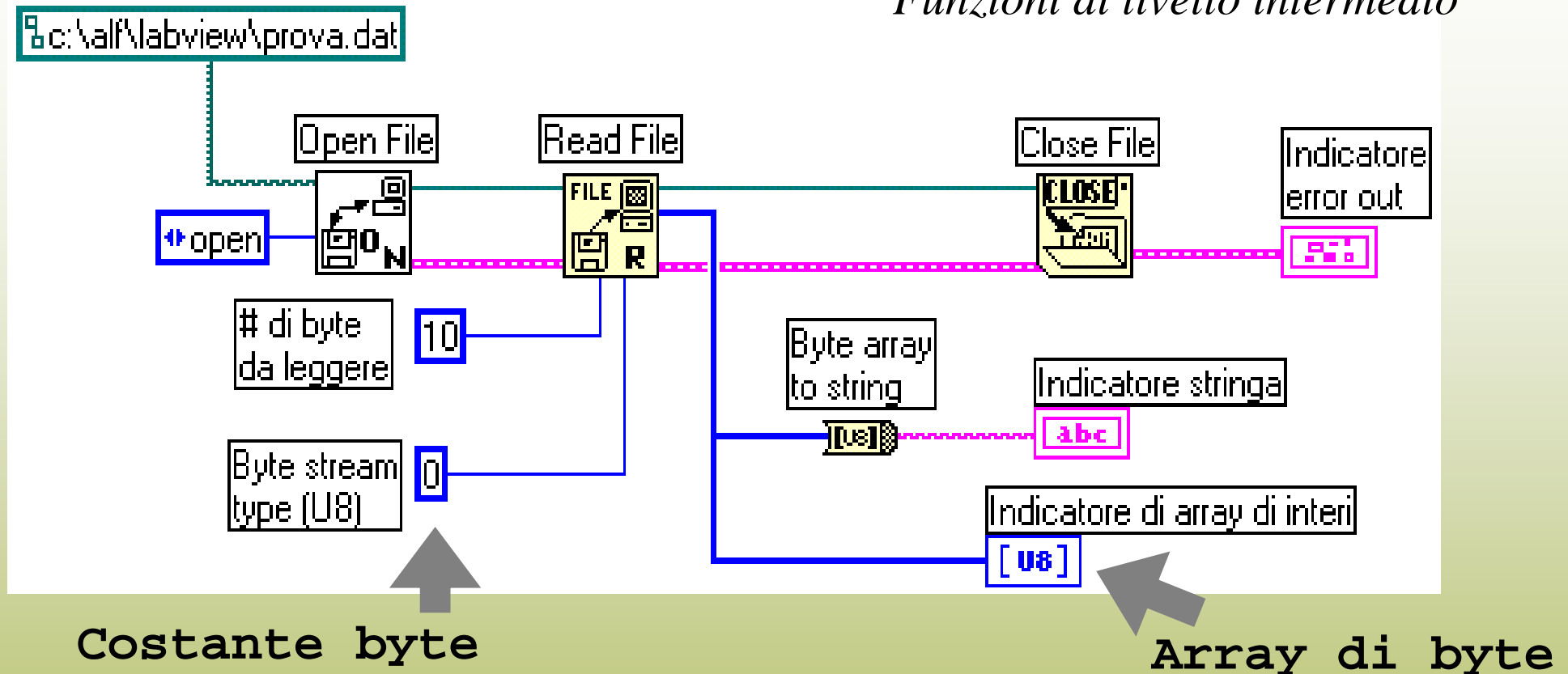
Indicatore stringa

"#&+2;FSbs"

Il contenuto di un file può avere diverse rappresentazioni

Lettura da file

Funzioni di livello intermedio



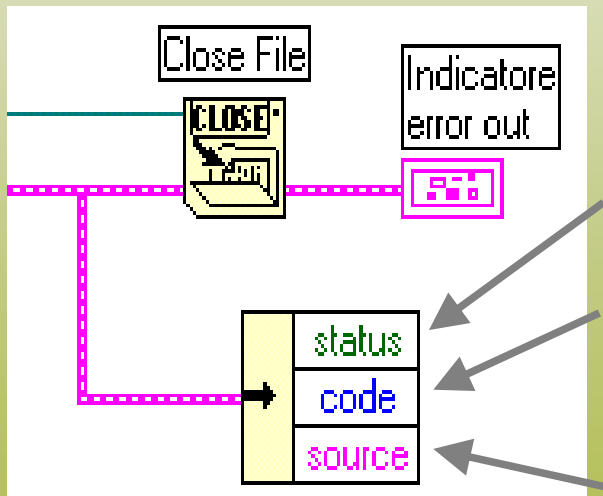
Il tipo di dato connesso al *Byte stream type* fissa il tipo degli elementi dell'array in uscita dalla *Read File*

Analogamente, la *Write File* è una funzione polimorfica

Gestione errori

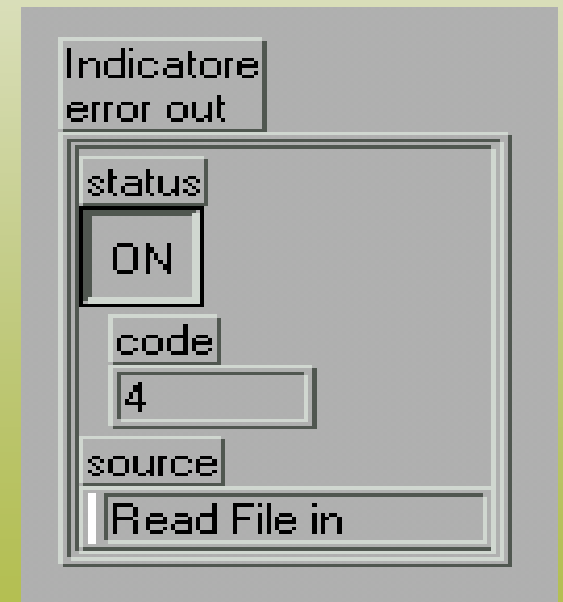
Funzioni di livello intermedio

- Non appena chiamata, ogni funzione controlla il connettore **error in**;
- Se trova **status = True**, significa che a monte si è verificato un errore: non esegue alcuna operazione e termina;
- Se si verifica un errore durante la sua esecuzione, **error out = True**.



Error Cluster:

- C'è stato errore?
- Numero associato all'errore
- In quale VI si è verificato?



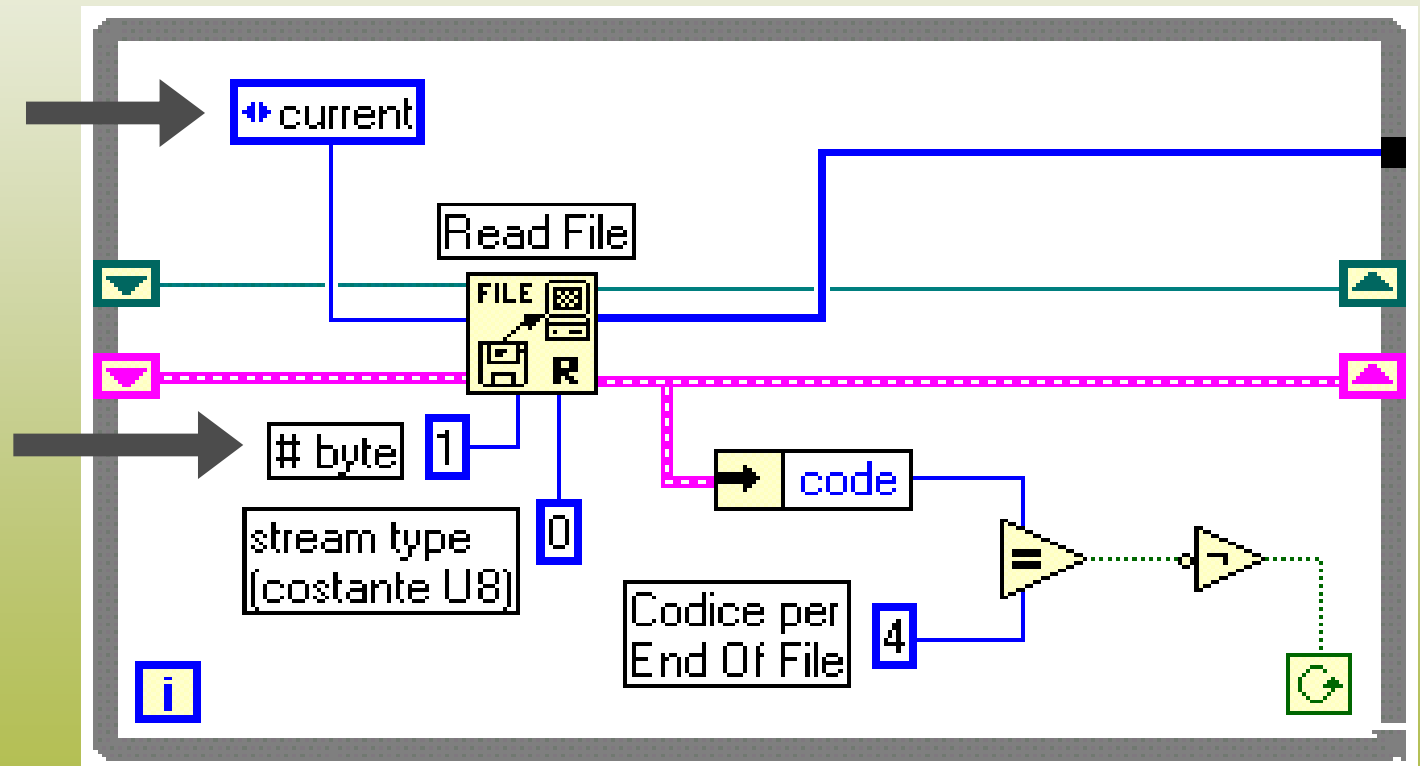
Lettura da file: condizione di fine file (EOF)

Funzioni di livello intermedio

E' possibile usare il campo **Code** del Cluster di errore per rilevare la condizione di raggiunta fine del file

Current: Legge il prossimo byte a partire dalla posizione corrente

Count: Numero di byte da leggere ad ogni chiamata



File di testo e file binari

```
a = 1.234e-5  
b = 200  
z=FALSE
```

Un file di testo è una sequenza di stringhe di caratteri ASCII delimitate da sequenze di *fine linea* (CR+LF):

a = 1.234e-5<CR><LF>b = 200<CR><LF>z=FALSE ...

- Facilmente interpretabile
- Scambio dati con altre applicazioni (fogli elettronici o *word processor*)
- La rappresentazione dei valori numerici è a lunghezza variabile
- L'accesso ai dati deve essere sequenziale

In un file binario la codifica dei dati dipende dalla particolare applicazione

- La rappresentazione dei valori numerici è a lunghezza fissa, e richiede un numero minore di byte
- L'accesso può essere casuale

File di tipo Foglio Elettronico (Spreadsheet)

- I dati sono organizzati in righe e colonne
- Le colonne sono separate da un delimitatore (TAB)
- Le righe sono separate da sequenze *new line* (CR+LF)

File:

```
0<TAB>0.4258<CR><LF>
1<TAB>0.3073<CR><LF>
2<TAB>0.9453<CR><LF>
3<TAB>0.964<CR><LF>
4<TAB>0.9517<CR><LF>
```

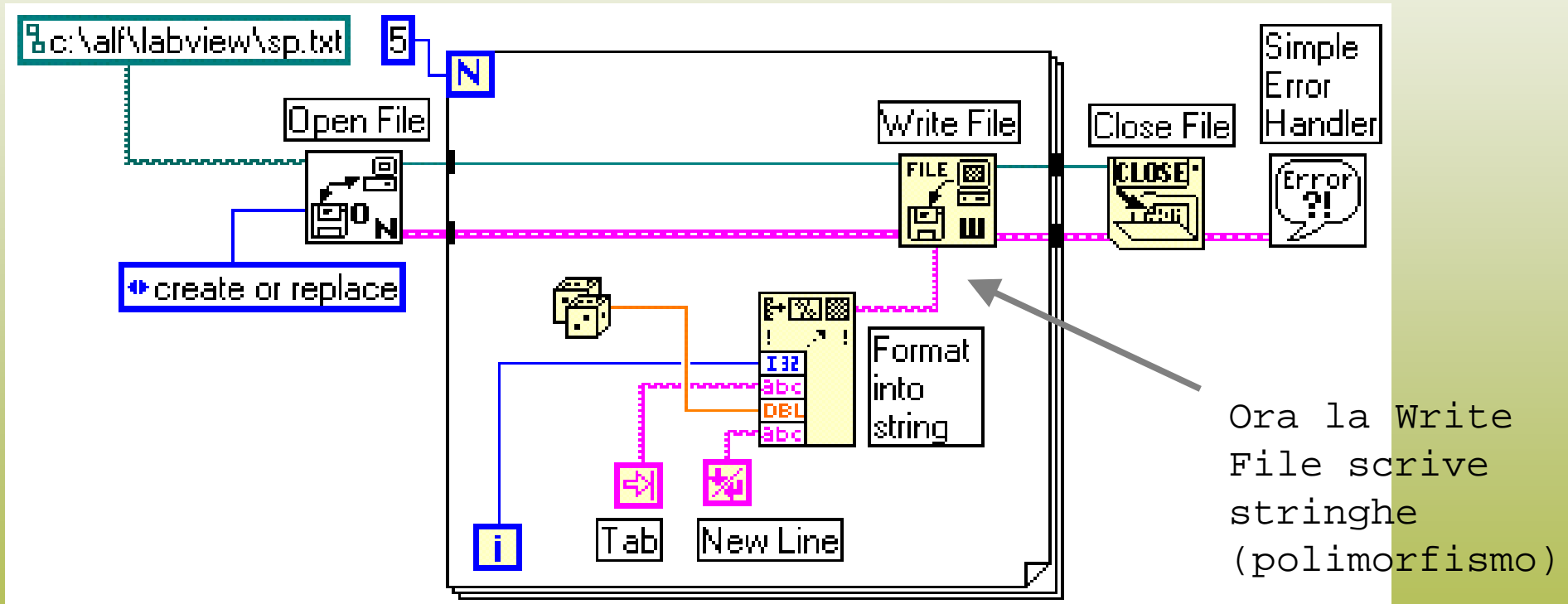
Nel foglio elettronico:

	A	B	C
1	0	0.4258	
2	1	0.3073	
3	2	0.9453	
4	3	0.964	
5	4	0.9517	
6			

Scrittura su spreadsheet

Con funzioni di livello intermedio

sp - Blocco note	
File	Modifica
Cerca	?
0	0,994535
1	0,424785
2	0,214862
3	0,104599
4	0,922959



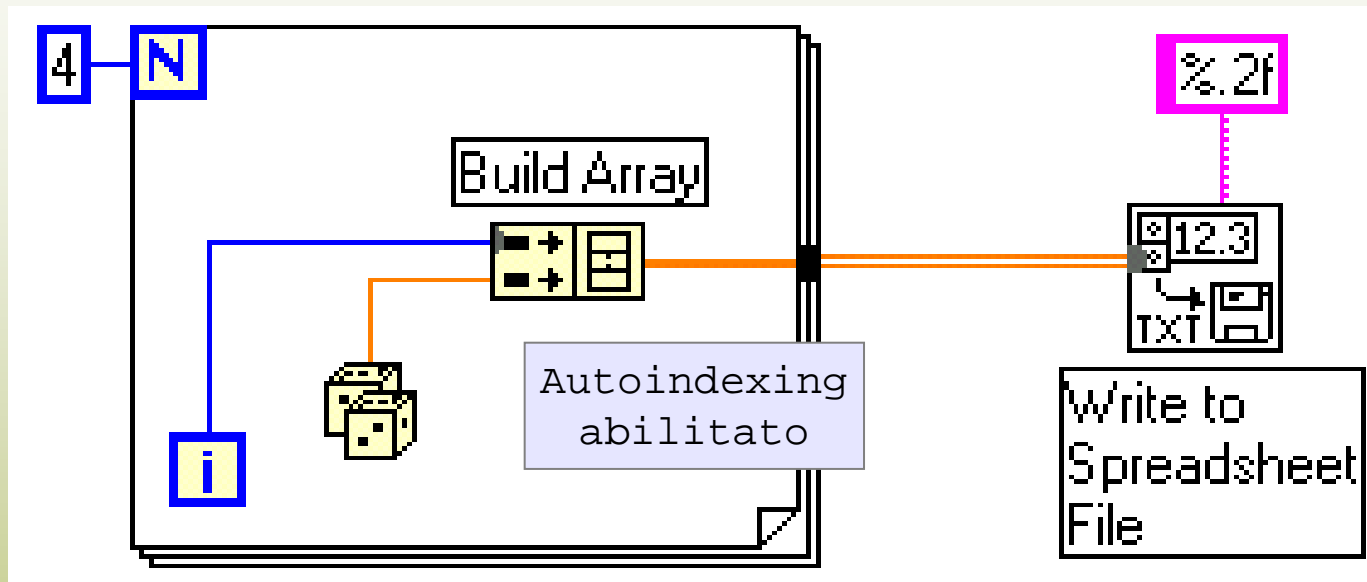
Funzioni di alto livello per i file

- **Write to Spreadsheet File**
- **Read from Spreadsheet File**
- **Write Characters to File**
- **Read Characters from File**
- **Read Lines from File**

- ✓ Effettuano direttamente le operazioni più frequenti di I/O su file di testo
- ✓ Aprono e chiudono il file (non deve farlo il chiamante)
- ✓ Nel loro diagramma a blocchi chiamano le funzioni di livello intermedio

Scrittura su spreadsheet

Con funzioni di alto livello



dati - Blocco n.		
File	Modifica	Ce
0,00	0,37	
1,00	0,21	
2,00	0,25	
3,00	0,57	

- La funzione *Write to Spreadsheet File* accetta in ingresso array 1D e 2D
- Se non specifica un nome di file, si apre una finestra di dialogo “Apri file..”
- Si può specificare il carattere delimitatore (TAB per default), trasporre l'array 2D, aggiungere i dati ad un file esistente (*append*)

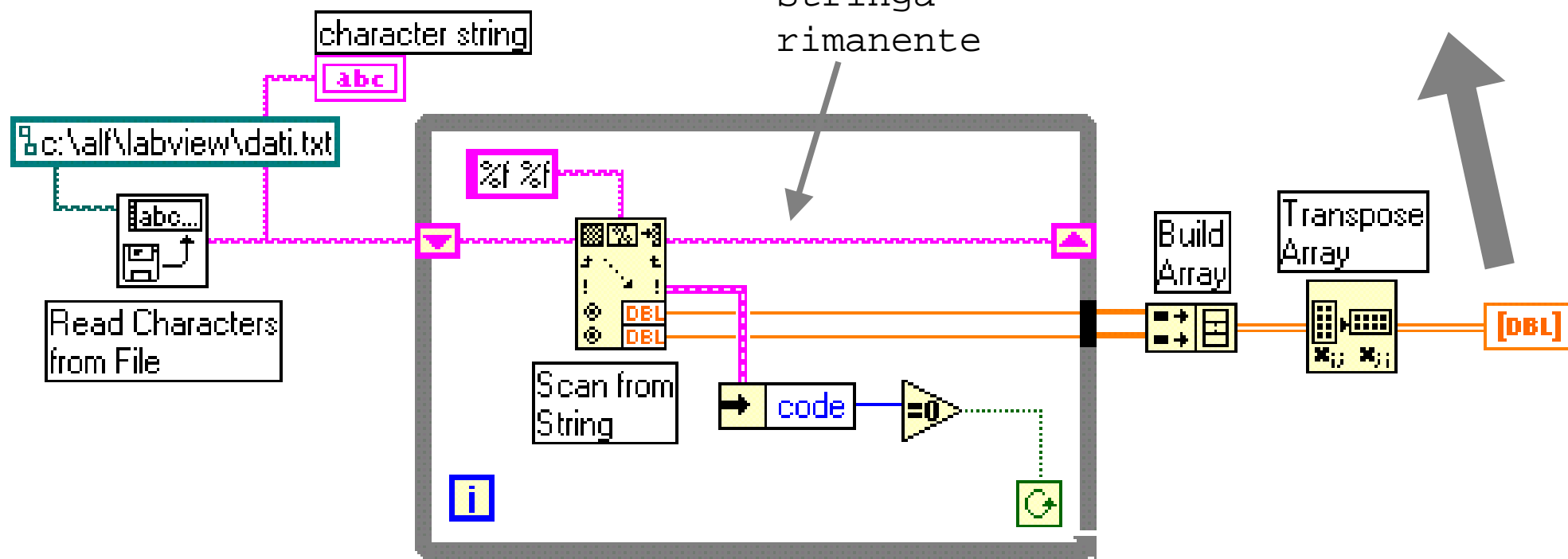
Lettura da spreadsheet

Con funzioni di livello "intermedio"

0,00\t0,35\r\n1,00\t0,19\r\n2,00\t0,28\r\n3,00\t0,29\r\n3,00\t0,00\r\n

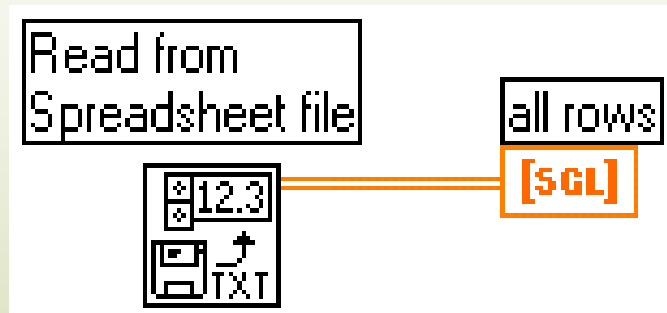
0	0,00	0,35
0	1,00	0,19
	2,00	0,28
	3,00	0,29
	3,00	0,00

Stringa
rimanente



Lettura da spreadsheet

Con funzioni di alto livello



all rows		
0	0,000	0,350
0	1,000	0,190
	2,000	0,280
	3,000	0,290

Altre funzioni di alto livello:

- *Write Characters to File*: Scrive una stringa di caratteri in un file
- *Read Lines from Files*: Legge un numero specificato di *linee* da un file
- *Binary File VIs*: 4 funzioni per la lettura/scrittura di array di **I16** o **SGL** su file

Variabili locali (*locals*)

Sono strutture che consentono di

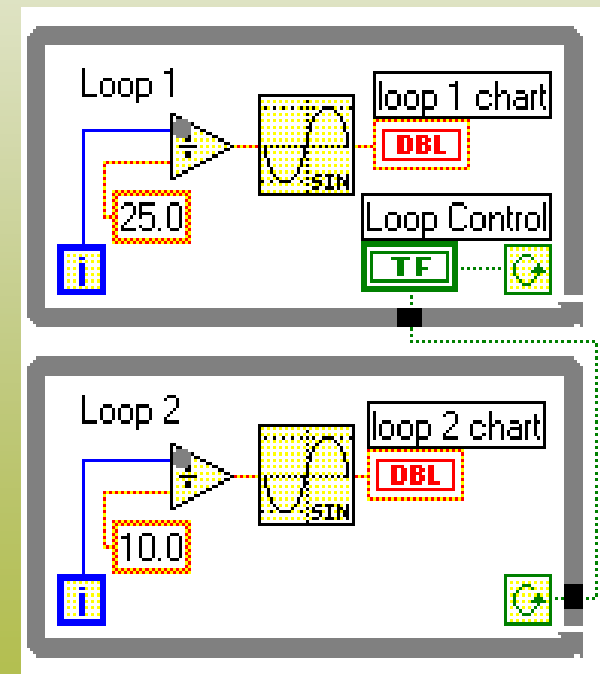
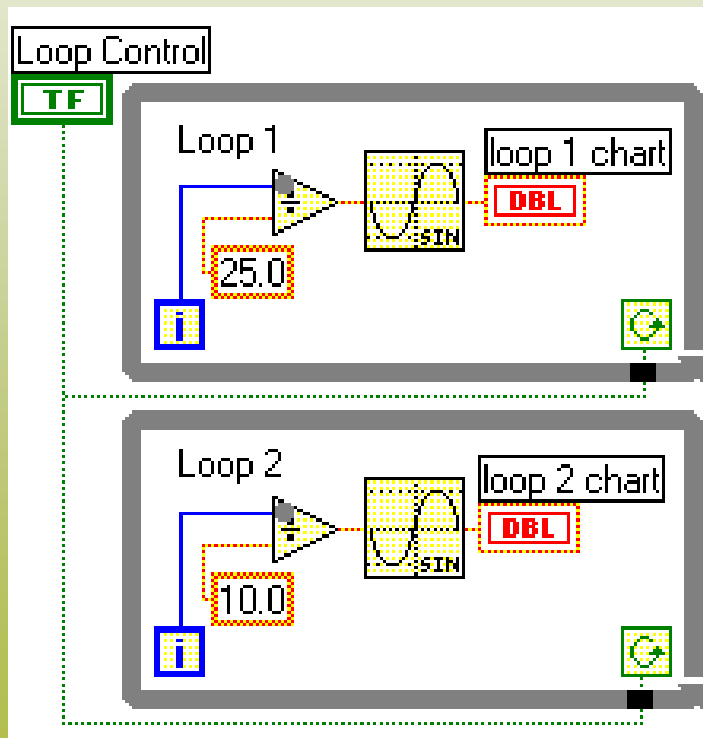
- Scrivere su un indicatore da più punti di un VI
- Leggere da un controllo da più punti di un VI
- Scrivere su controlli
- Leggere da indicatori

Esempio

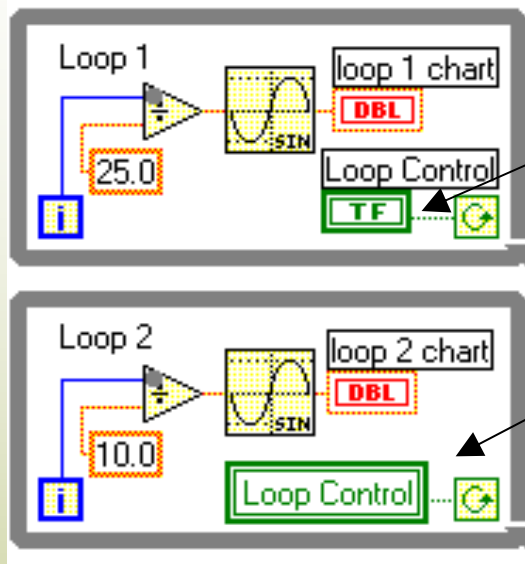
In un VI sono presenti due cicli *while* eseguiti parallelamente

Si vogliono fermare entrambi alla pressione di un singolo bottone

Soluzioni sbagliate:



Soluzione corretta:



Terminale del controllo (tasto “stop”)

Variabile locale

- configurata come *read local* (sorgente dati)
- associata al controllo

Come creare una variabile locale:

- Tasto destro del mouse sul terminale e “Create >> Local Variable”
oppure
- Dalla palette funzioni “Structures >> Local Variable”; selezionare poi (tasto destro sulla variabile) con “Item Select” il controllo/indicatore

Infine, scegliere se si vuole leggere o scrivere da/verso la variabile

Note sulle variabili locali

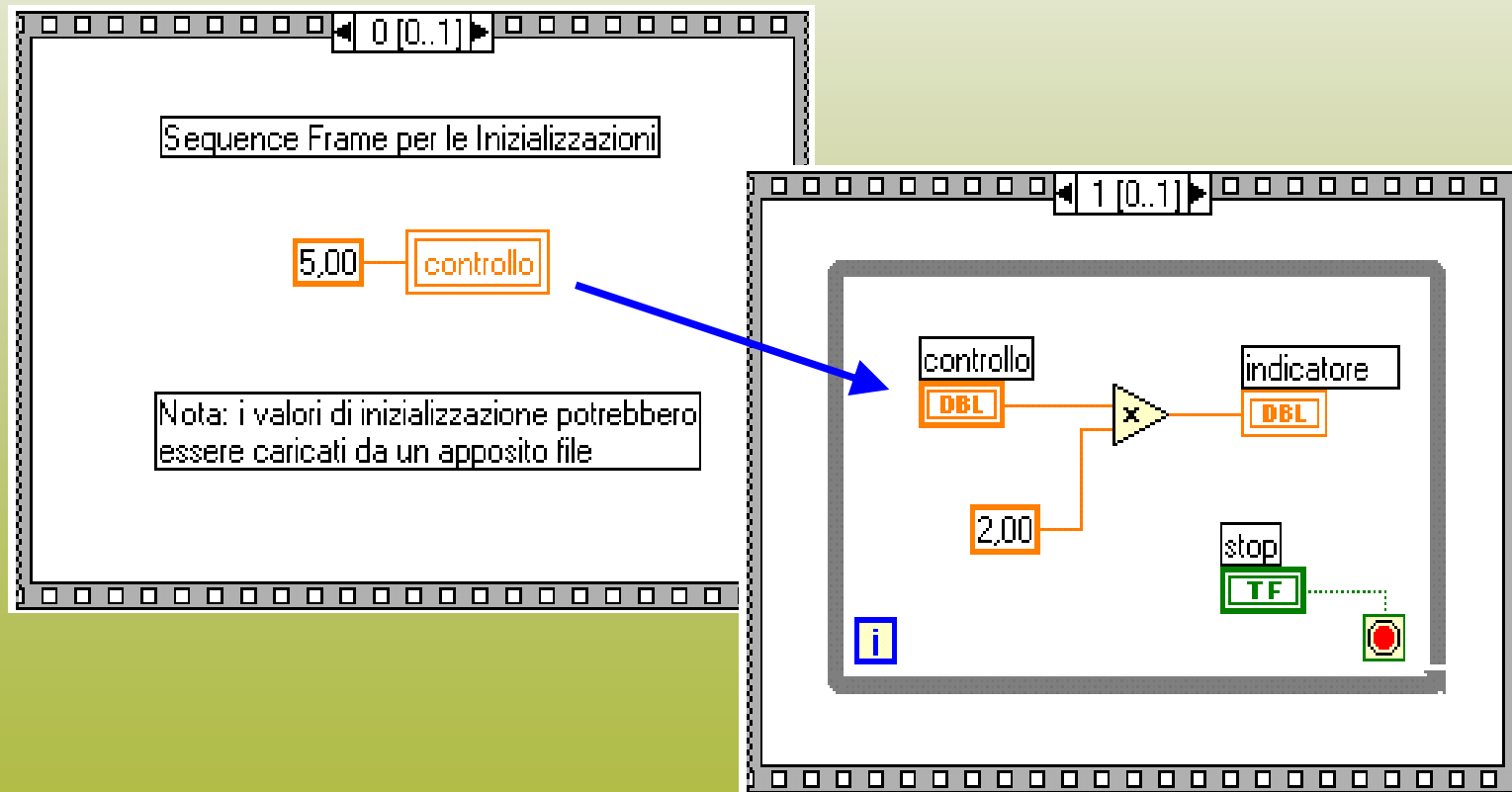
Un controllo/indicatore al quale si vuole associare una variabile locale **deve** avere una label, che diventerà il nome della variabile stessa

La scrittura su una *local* aggiorna il corrispondente controllo/indicatore

La lettura da una *local* legge il valore corrente del corrispondente controllo/indicatore

Inizializzazione di controlli

- ✓ Tramite le *locals* è possibile dare valori iniziali ai controlli
- ✓ Le inizializzazioni di tutti i controlli possono essere raccolte in una sezione apposita del codice (*startup*)
- ✓ Se necessario, i valori di inizializzazione si possono caricare da file. Una *sub-palette* di funzioni (“Configuration file”) permette la gestione di file di configurazione



Variabili globali (*globals*)

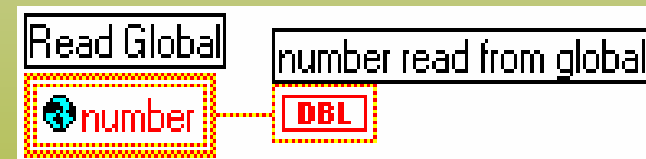
Le variabili globali hanno usi simili alle *locals*, ma sono accessibili da altri VI

Impieghi:

- ✓ Controllo dell'esecuzione di uno o più VI da un altro VI
- ✓ Condivisione di dati tra più VI
- ✓ Sono un tipo particolare di VI:
- ✓ I dati sono memorizzati negli elementi del pannello
- ✓ Non hanno diagramma a blocchi



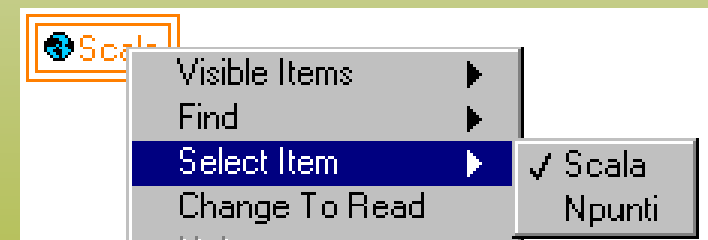
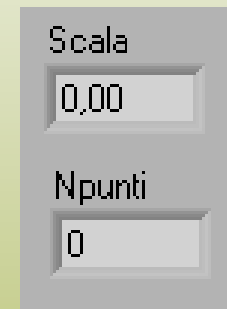
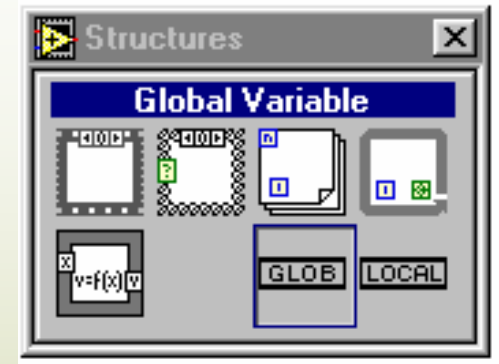
Write Global



Read Global

Creare ed usare variabili globali

- ✓ *Palette* “Structures >> Global Variable”
- ✓ Dal suo menu di contesto “Open Front Panel”
- ✓ Inserire nel pannello gli elementi necessari (*label* obbligatoria)
- ✓ Salvare il VI - variabile globale
- ✓ Nel diagramma del VI di partenza, dal menu di contesto della variabile globale: “Select Item” per scegliere l'elemento
- ✓ Da altri VI, la variabile globale è accessibile tramite la voce “Select a VI...” della *palette* delle funzioni



Note sull'uso di variabili locali e globali

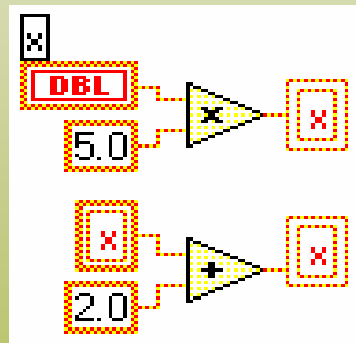
Inizializzare ogni variabile globale (scrivendovi) prima di leggerne il contenuto. In assenza di inizializzazione, sarà restituito un valore di *default*.

Le variabili locali e globali sono un'eccezione al *dataflow programming*

Rendono il diagramma più difficile da capire

L'accesso ai dati in una variabile è più lento

Attenzione ai casi di *race condition*!



Risultato

$x = x * 5$

$x = x + 2$

oppure

$x + 2$

$x = x * 5$

- Usare le variabili solo quando non esistono altre possibilità

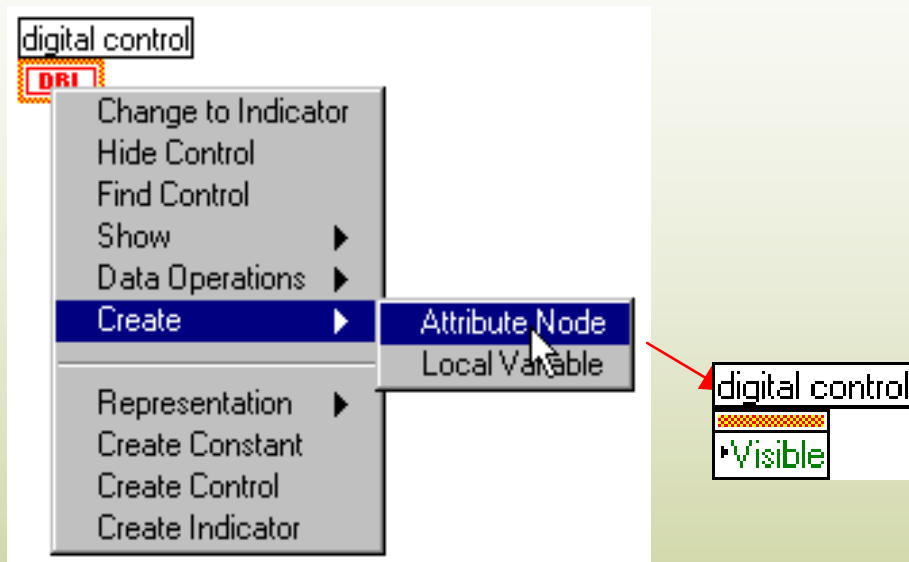
Property nodes (o attribute nodes)

Consentono di accedere in lettura e scrittura alle proprietà di un elemento del pannello (non al contenuto)

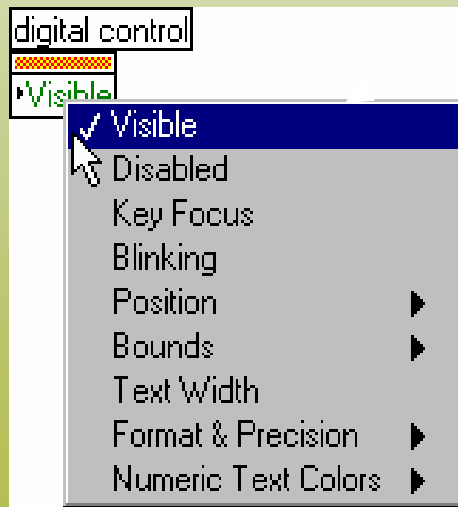
Esempi:

- Il colore del testo di un indicatore numerico
- La condizione di “abilitato/disabilitato” di un bottone
- Le voci di un *ring control*
- Scale e cursori di un *chart*
- *Posizione e dimensioni di un controllo/indicatore*

Creazione di un *property node*

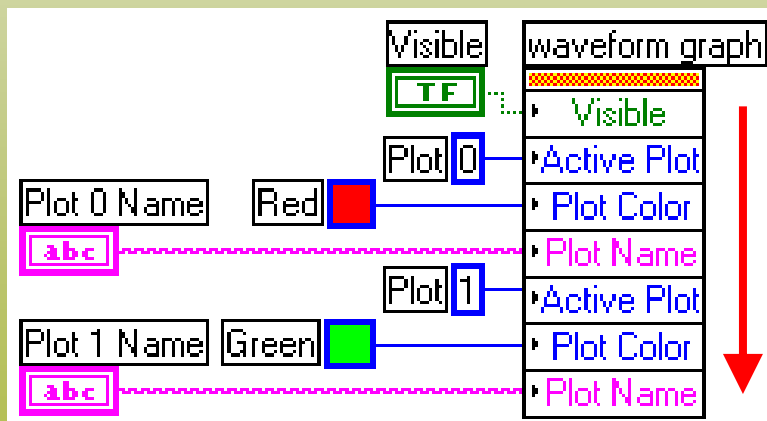
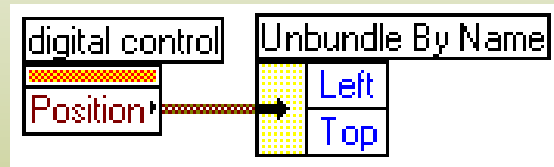
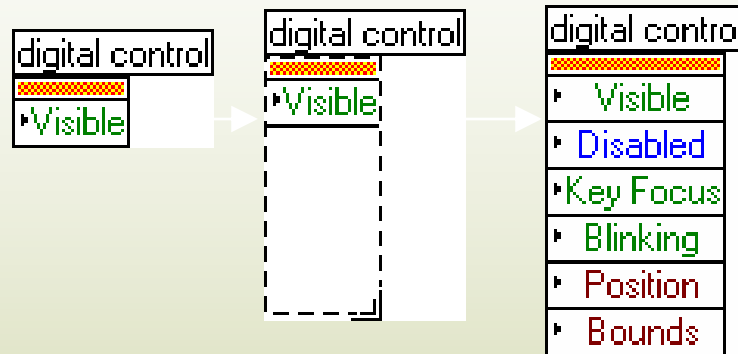


- Dal menu di contesto di un elemento del pannello o del suo terminale:
- “Create >> Attribute Node”



- Usare l'*operating tool* per scegliere l'attributo

Usare i *property nodes*



- ✓ Scegliere (tasto destro) se leggere o scrivere da/verso l'attributo
- ✓ Altre proprietà si possono aggiungere ridimensionando il *property node*
- ✓ Il colore aiuta ad individuare il tipo di dato
- ✓ Alcune proprietà sono *clusters* - usare Bundle e Unbundle
- ✓ Ordine di valutazione degli attributi: dall'alto verso il basso
- ✓ Usare la *help window* (Ctrl-H) per avere informazioni su un attributo

Attributi comuni di controlli/indicatori

Visible	è visibile
Disabled	se = 2 non è modificabile dall'utente (aspetto inalterato) se = 1 è non modificabile e grigio se = 0 è modificabile
Key Focus	possiede il cursore
Position	<i>cluster</i> posizione (<i>left</i> e <i>top</i>): pixel dall'angolo in alto a sx
Blinking	lampeggio intermittente
Format	tipo enumerato: decimale, scientifico, ...
Precision	numero di cifre decimali
...	

Inoltre, possono essere lette/modificate tutte le stringhe di un controllo/indicatore (*label,caption,...*) ed i colori delle sue varie parti

L'elenco degli attributi cambia a seconda del tipo di controllo/indicatore

Gestione di un menu

