Afficher : ☑ Delphi ☑ C++
Préférences d'affichage

getch

De RAD Studio

Remonter à Conio.h - Index

Header File

conio.h

Category

Console I/O Routines

Prototype

int getch(void);

Description

Gets character from keyboard, does not echo to screen.

getch reads a single character directly from the keyboard, without echoing to the screen.

> **Remarque :** Do not use this function in Win32 GUI applications.

Return Value

getch returns the character read from the keyboard.

Example

```
#include <conio.h>
#include <stdio.h>
int main(void)
{
  int c;
  int extended = 0;
  c = getch();
  if (!c)
    extended = getch();
  if (extended)
    printf("The character is extended\n");
  else
    printf("The character isn't extended\n");
  return 0;
}
```

Portability

POSIX	Win32	ANSI C	ANSI C++
	+		

Catégories :

- Référence C++
- C++

- Envoyer mon commentaire
-
- Page Wiki en cours

Afficher : ☑ Delphi ☑ C++
Préférences d'affichage

getchar, getwchar

De RAD Studio

Remonter à Stdio.h - Index

Header File

stdio.h

Category

Console I/O Routines

Prototype

int getchar(void);

wint_t getwchar(void);

Description

Gets character from stdin.

getchar is a macro that returns the next character on the named input stream stdin. It is defined to be getc(stdin).

> **Remarque :** Do not use this function in Win32 GUI applications.

Return Value

On success, getchar returns the character read, after converting it to an **int** without sign extension.

On end-of-file or error, it returns EOF.

Example

```
#include <stdio.h>
int main(void)
{
   int c;
/*
Note that getchar reads from stdin and is line buffered; this means it will not return until you press ENTER.
 */
   while ((c = getchar()) != '\n')
      printf("%c", c);
   return 0;
}
```

Portability

	POSIX	Win32	ANSI C	ANSI C++
getchar	+	+	+	+
getwchar		+	+	+

Catégories :

- Référence C++
- C++

- Envoyer mon commentaire
-
- Page Wiki en cours

Afficher : ☑ Delphi ☑ C++
Préférences d'affichage

getche

De RAD Studio

Remonter à Conio.h - Index

Header File

conio.h

Category

Console I/O Routines

Prototype

int getche(void);

Description

Gets character from the keyboard, echoes to screen.

getche reads a single character from the keyboard and echoes it to the current text window using direct video or BIOS.

> **Remarque :** Do not use this function in Win32 GUI applications.

Return Value

getche returns the character read from the keyboard.

Example

```
#include <stdio.h>
#include <conio.h>
int main(void)
{
   char ch;
   printf("Input a character:");
   ch = getche();
   printf("\nYou input a '%c'\n", ch);
   return 0;
}
```

Portability

POSIX	Win32	ANSI C	ANSI C++
	+		

Catégories :

- Référence C++
- C++

- Envoyer mon commentaire

- Page Wiki en cours

Afficher : ☑ Delphi ☑ C++
Préférences d'affichage

kbhit

De RAD Studio

Remonter à Conio.h - Index

Header File

conio.h

Category

Console I/O Routines

Prototype

int kbhit(void);

Description

Checks for currently available keystrokes.

kbhit checks to see if a keystroke is currently available. Any available keystrokes can be retrieved with getch or getche.

> **Remarque :** Do not use this function in Win32 GUI applications.

Return Value

If a keystroke is available, kbhit returns a nonzero value. Otherwise, it returns 0.

Example

```
#include <conio.h>
int main(void)
{
 cprintf("Press any key to continue:");
    while (!kbhit()) /* do nothing */ ;
    cprintf("\r\nA key was pressed...\r\n");
    return 0;
}
```

Portability

POSIX	Win32	ANSI C	ANSI C++
	+		

Catégories :

- Référence C++
- C++

- Envoyer mon commentaire
-
- Page Wiki en cours

Afficher : ☑ Delphi ☑ C++
Préférences d'affichage

scanf, wscanf

De RAD Studio

Remonter à Stdio.h - Index

Header File

stdio.h

Category

Console I/O Routines

Prototype

```
int scanf(const char *format[, address, ...]);

int wscanf(const wchar_t *format[, address, ...]);
```

Description

Scans and formats input from the `stdin` stream.

> **Note:** For Win32 and Win64 GUI applications, `stdin` must be redirected.

The `scanf` function:

- Scans a series of input fields one character at a time.
- Formats each field according to a corresponding format specifier passed in the format string `*format`.
- `vsscanf` scans and formats input from a string, using an argument list.

There must be one format specifier and address for each input field.

`scanf` might stop scanning a particular field before it reaches the normal end-of-field (whitespace) character, or it might terminate entirely. For details about why this might happen, see When ...scanf Stops Scanning.

> **Note:** `scanf` can lead to unexpected results if you diverge from an expected pattern.
> You must provide information that tells `scanf` how to synchronize at the end of a line.

The combination of gets or fgets followed by sscanf is safe and easy, and therefore recommended over `scanf`.

Return Value

On success, `scanf` returns the number of input fields successfully scanned, converted, and stored.

The return value does not include scanned fields that were not stored.

On error:

- If no fields were stored, **scanf** returns 0.
- If **scanf** attempts to read at end-of-file or at end-of-string, it returns EOF.

More About `scanf`

Example

```
#include <stdio.h>
int main(void)
{
   char label[20];
   char name[20];
   int entries = 0;
   int loop, age;
   double salary;
   struct Entry_struct
   {
      char  name[20];
      int   age;
      float salary;
   } entry[20];
/* Input a label as a string of characters restricting to 20 characters */
   printf("\n\nPlease enter a label for the chart: ");
   scanf("%20s", label);
   fflush(stdin);  /* flush the input stream in case of bad input */
/* Input number of entries as an integer */
  printf("How many entries will there be? (less than 20) ");
  scanf("%d", &entries);
  fflush(stdin);   /* flush the input stream in case of bad input */
/* input a name restricting input to only letters uppercase or lowercase */
   for (loop=0;loop<entries;++loop)
   {
      printf("Entry %d\n", loop);
      printf("  Name   : ");
      scanf("%[A-Za-z]", entry[loop].name);
      fflush(stdin);  /* flush the input stream in case of bad input */
/* input an age as an integer */
      printf("  Age    : ");
      scanf("%d", &entry[loop].age);
      fflush(stdin);  /* flush the input stream in case of bad input */
/* input a salary as a float */
      printf("  Salary : ");
      scanf("%f", &entry[loop].salary);
      fflush(stdin); /* flush the input stream in case of bad input */
   }
/* Input a name, age, and salary as a string, integer, and double */
   printf("\nPlease enter your name, age and salary\n");
   scanf("%20s %d %lf", name, &age, &salary);
/* Print out the data that was input */
   printf("\n\nTable %s\n",label);
   printf("Compiled by %s  age %d  $%15.2lf\n", name, age, salary);
   printf("-----------------------------------------------------\n");
   for (loop=0;loop<entries;++loop)
      printf("%4d | %-20s | %5d | %15.2lf\n",
         loop + 1,
         entry[loop].name,
         entry[loop].age,
         entry[loop].salary);
   printf("-------------------------------------------------------\n");
   return 0;
}
```

See Also

- scanf Format Specifiers
- scanf Format String
- Input Fields for scanf functions
- When ...scanf Stops Scanning
- Format Specifier Conventions
- Format Specifiers in C/C++
- Extended Types Formatted I/O
- Unicode Input Format Specifiers
- Argument Suppression
- Pointer-size and Argument-type Modifiers
- gets, _getws
- fgets, fgetws

Catégories :

- Référence C++
- C++

- Envoyer mon commentaire
-
- Page Wiki en cours