

The C-Language Keywords and Symbols

Keywords		Symbols			
MEMORY	CONTROL	CONTROL	LOGIC	MATH	BIT OP
01.void	21.return	#	==	*	
02.char	22.if	< >	!=	%	&
03.int	23.else	//	<	/	^
04.short	24.switch	/* */	>	+	~
05.long	25.case	()	<=	-	<<
06.float	26.default	{ }	>=		>>
07.double	27.while	;	&&		
08.signed	28.do	,			
09.unsigned	29.for	"	!		
10.struct	30.break	'			
11.union	31.continue	=			
12.enum	32.goto	[]			
13.const		:			
14.volatile		?			
15.auto		.			
16.extern		\			
17.static		MEMORY			
18.register		&			
19.typedef		*			
20.sizeof					

switch - case - default

The "switch/case" Keyword allows us to run instruction codes based on a SOURCE NUMBER and a MATCHING NUMBER (for selective execution)

"switch" Keyword used alone

Part1: "switch" Keyword

Part2: switch_number - Source Number, any whole numbers(no fractions)

Part3: switch_body

Part4: "case" Keyword

Part5: case_number - Matching Number, any whole numbers(no fractions)

Part6: "break" Keyword

Part7: "default" Keyword

Part8: "break" Keyword

Part1: "switch" Keyword

Part2: switch_number

- placed within the bracket () pair

Part3: switch_body

- within a Curly Bracket { } pair

Part5: case_number

- followed by colon :

Part4: "case" Keyword

- followed by space

Part6: "break" Keyword

- followed by semi-colon ;

Part7: "default" Keyword

- followed by colon ;

Part8: "break" Keyword

- followed by semi-colon ;

switch(switch_number)

{

case case_number:

instruction code;

break;

default:

instruction code;

break;

}

zero
or
many

zero
or
many

switch - case - default (Step by Step)

STEP 1/4:

Make a skeleton switch-case-default structure

```
switch(0) {
    default:
    break;
}
```

STEP 2/4:

switch/switch_number - a whole number(no fractions)

```
int switch_number; // a Variable with any whole number
switch(switch_number) { // switch_number = any whole number(no fractions)
    default:
    break;
}
```

STEP 3/4:

case/case_number to match switch_number

```
int switch_number; // a Variable with any whole number
switch(switch_number) {
    case case_number: // case_number = any whole number to match switch_number
    instruction_code;
    break;
    default:
    break;
}
```

- we can have **zero or many "case"** structure within the "switch" body
- when the **switch_number** and the **case_number matches**, the **instruction_code** after the **case colon : symbol** will start to run until it encounters "break;" where it will exit the switch (after the closing curly bracket })

STEP 4/4:

default

```
int switch_number; // a Variable with any whole number
switch(switch_number) {
    case case_number: // case_number = any whole number to match switch_number
    instruction_code;
    break;
    default:
    instruction_code;
    break;
}
```

- we can have only **one "default"** structure within the "switch" body
- when the **switch_number** does not match any **case_number matches**, the **instruction_code** after the **default colon : symbol** will start to run until it encounters "break;" where it will exit the switch (after the closing curly bracket })

- **break** Keyword

- "break;" will jump to the code after the body closing curly bracket } for the following structures: switch , while , do and for

Arduino IDE|Save PROGRAM as: **c_switch**

Enter codes below and upload. Use the Serial Monitor to see results

```
void setup() {
  Serial.begin(9600);Serial.print("\n\nSerial Monitor(9600)...");

int menu_options;
  // menu_options Variable value will change via different methods
  // tactile buttons, keyboard, sensors, etc
  menu_options = 2; // dummy test value, change to test

  Serial.print("\n\nBefore entering the switch-case-default");

  switch(menu_options) {
    case 1:
      Serial.print("\n\nPROGRAM continues here,");
      Serial.print("\nmenu_options Variable matches case 1:");
      break;
    case 2:
      Serial.print("\n\nPROGRAM continues here,");
      Serial.print("\nmenu_options Variable matches case 2:");
      break;
    default:
      Serial.print("\n\nUnknown Menu Option");
      break;
  }
  Serial.print("\n\nEnd of switch-case-default");
}
void loop(){}
}
```

In this example, the menu_option have a value of 2; since the case 2: matches this value, the codes in the case 2: will be executed until break; where it exits the switch structure. It will then run the codes after the closing switch body closing curly bracket }

Try change the menu_option value with different number to see different effects