Arduino a cyklus (čo je to) ? For, While, Do-while :)

Už z názvu sme schopní odvodiť, že cyklus je niečo, čo sa opakuje v nejakej slučke.

Prečo sa používajú cykly?

Cykly slúžia nielen k lepšej prehľadnosti kódu. Programy s cyklami pracujú efektívnejšie, ako keby sme kód písali viackrát za sebou.

Aké cykly existujú?

Existujú tri druhy cyklov:

- for,
- while,
- do-while.

Podrobne si vysvetlíme, ako ktorý cyklus funguje.

Prvý a najzákladnejší cyklus, ktorý si ukážeme bude **for**.

Tento cyklus je používaný najčastejšie, hlavne pre jeho prehľadnosť, jednoduchosť a vopred určený počet priechodov.

Počas cyklu for sa mení postupne hodnota i, kým nenadobudne takú, ktorú si prajeme. Cyklus for sa skladá z troch hlavných častí:

- premenná,
- podmienka,
- príkaz.

Syntax takéhoto cyklu vyzerá nasledovne:

```
1  void setup()
2  {
3    Serial.begin(9600);
4    //for cyklus - vložíme ho do setupu z toho dôvodu, aby sa nám cyklus stále neopakoval
5    for (int i = 0; i < 4; i++)
6      {
7        Serial.prinln("výpis pomocou for cyklu"); //príkaz na vypisovanie textu pomocou for cyklu (text sa vypíše 4x)
8      }
9  }
10
11 void loop()
12 {
13
14 }
```

Program s ukážkou cyklu for

Premenná (int i = 0) - v tomto prípade vlastne určuje počiatočnú hodnotu cyklu. Najčastejšie to býva nula, pretože od tej sa najčastejšie začína. Premenné vo for cykloch sa väčšinou značia písmenom i (index), ale pomenovanie je na vás.

Podmienka (i < 4) - nám udáva v akom prípade sa vykoná ďalší krok nášho for cyklu. Keď nastane situácia, že nami vytvorená podmienka platiť nebude, cyklus skončí a program pokračuje ďalej. V našom prípade for cyklus prestane fungovať, keď premenná i bude väčšia, než číslo 4.

Príkaz (i++) - tu hovorí, čo sa má s hlavnou premennou v cykle stať - konkrétne či sa má zvýšiť, alebo znížiť. K tomu využijeme operátorov ++ (inkrementácia) a -- (dekrementácia). Tieto operátory samozrejme môžeme využívať aj mimo definície cyklu for, ich funkčnosť zostane rovnaká. V našom prípade zvýšime vždy po každom cykle premennú i o číslo 1.

Ako môžeme vidieť, syntax for cyklus je prehľadná a tiež jednoduchá. Určite je to lepšia voľba, než kód nezmyselne viackrát opisovať.

Ďalší cyklus, ktorý si vysvetlíme je **while**.

Ten funguje na trochu inom princípe, než cyklus for. Išlo by si význam tiež odvodiť z slovenského prekladu, kedy while znamená „zatiaľ čo“. Tento cyklus nám teda hovorí, že príkaz uvedený v bloku sa opakuje pokiaľ platí podmienka.

Copyright © 2013 - 2025 Encyklopédia poznania

Syntax while je vlastne ešte jednoduchšie, než u for. V zátvorke vedľa while je podmienka a pod ňou už je len zátvorkami označený blok, v ktorom sú všetky príkazy. Cez cyklus while je možné urobiť aj cyklus for, pretože for je vlastne špeciálny prípad while.

While cyklus sa ale často používa na trochu iné veci, ktorých príklad si tiež ukážeme. V praxi môže využitie tohto cyklu vyzerať nasledovne:

```
bool stav_cyklus = true ; //premenná, ktorá môže mať 2 stavy - true/false ---> určuje nám funkciu while cyklu

void setup()
{
  Serial.begin(9600) ;
  //while cyklus, ak sa tu nachádza podmienka ktorá bude platiť, tak cyklus bude prebiehať (v našom prípade neskončí)
  while(stav_cyklus == true)
   {
    Serial.println("while cyklus prebieha") ; //výpis do sériového monitoru
   }
}

void loop()
{
}
```

Program s ukážkou cyklu while

Mali by ste vedieť:

Môžete vidieť, že náš cyklus vkladáme do funkcie void setup(), je to z toho dôvodu, aby sme si ukázali funkčnosť samotného while cyklu. Keby sme cyklus vložili do funkcie void loop(), tak neuvidíme žiadny rozdiel. Je to z toho dôvodu, pretože void loop() je tiež cyklus a neustále sa opakuje.

Posledným cyklom, ktorý si dnes spomenieme je cyklus **do-while**.

Tento cyklus je veľmi podobný predošlému while cyklu. Zásadný rozdiel medzi nimi je ale ten, že v cykle do-while je podmienka umiestnená až na konci cyklu. Tým máme zaistené, že sa ľubovoľný cyklus vykoná aspoň raz.

Syntax do-while je tiež pomerne jednoduchý, prvá časť cyklu je do, ktorý sa, ako už sme si povedali vykoná minimálne jedenkrát. Po príkaze alebo bloku príkazov v cykle nasleduje na konci kľúčové slovo while s podmienkou, ktorá je po vykonaní akcie skontrolovaná. Keď sa podmienka vyhodnotí ako neplatná, je cyklus preskočený. Pokiaľ je podmienka platná, cyklus sa opakuje rovnako ako pri while.

Rovnako ako pri predchádzajúcich cykloch, aj tento si teraz ukážeme v praxi. Tu môžeme vidieť využitie do-while cyklu:

```
bool stav_cyklus = true ; //premenná, ktorá môže mať 2 stavy - true/false --> určuje nám funkciu while cyklu

//funkcia setup() - prebehne raz, potom sa už neopakuje
void setup()
{
    Serial.begin(9600) ; //Baudova rýchlosť - nastavíme na 9600 --> slúži pre komunikáciu na sériovom monitore
    do
      {
       Serial.println("Do-while cyklus prebieha") ; //výpis textu do sériového monitora
      }
    //do-while cyklus - prebehne raz a potom sa skontroluje, či platí podmienka, ak nie --> cyklus sa preskočí
    while (stav_cyklus == true) ;
}

//funkcia loop() - neustále se opakuje
void loop()
{
}
```

Program s ukážkou cyklu do-while

Hoci cyklus do-while nie je tak používaný ako while, alebo for, v určitých situáciách môže byť jeho použitie vhodnejšie.

Zdroje

Prevzaté a upravené z:

- https://www.itnetwork.sk/hardver-pc/arduino/programovaci-jazyk/cykly-a-ich-syntaxe-a-pouzitie.

Copyright © 2013 - 2025 Encyklopédia poznania