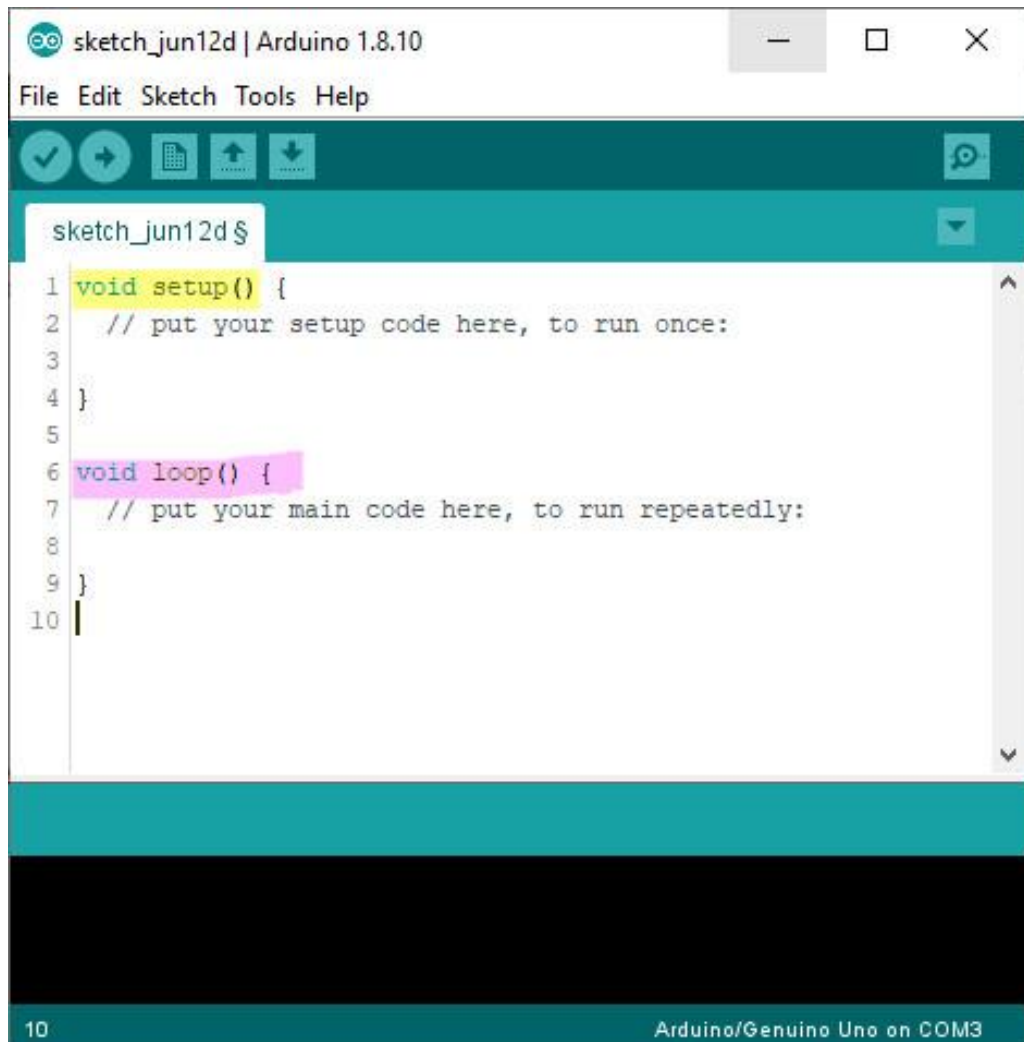


## بخش اول Structure

دو بخش اصلی در این قسمت نهفته است، که با هر بار ورود به نرم افزار آردوینو و ایجاد یک تب جدید برای شروع برنامه نویسی با آن رو به رو می‌شوید.

- Setup () Function
- Loop () Function
- 



```
sketch_jun12d | Arduino 1.8.10
File Edit Sketch Tools Help
sketch_jun12d $
1 void setup() {
2   // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7   // put your main code here, to run repeatedly:
8
9 }
10
```

10 Arduino/Genuino Uno on COM3

## معرفی voidsetup ()

در `voidsetup` توابع اصلی برای شروع برنامه نویسی تعریف می‌شود. از این تابع برای معرفی متغیرها، مدهای پایه، استفاده از دستورات کتابخانه ای استفاده می‌شود. فانکشن هایی که در `setup` تعریف می‌شود فقط یک بار اجرا می‌شود. از بخش های مهم دیگری که در `voidsetup` تعریف می‌شود، ورودی `Input` و خروجی `Output` است. در نهایت در این بخش از `Return` هم استفاده می‌شود. برنامه نوشته شده در این بخش با باز و بسته شدن سریال مانیتور و یا ریست توسط کلید قرار گرفته شده با آردوینو مجدد از اول اجرا می‌شود.

```
} ( ) setup void
: put your setup code here, to run once //

{
```

## معرفی Voidloop ()

زمانیکه دستورات نوشته شده در voidsetup اجرا شد، نوبت به voidloop یا همان توابع حلقه در برنامه می‌رسد. تا زمانیکه حلقه در برنامه تعریف شده باشد، تمامی دستورات نوشته شده اجرا می‌شود. تنها با قطع ارتباط و یا ریست برنامه متوقف خواهد شد. دستورات در تابع حلقه از بالا به پایین به صورت پیوسته اجرا می‌شود. به عنوان مثال اگر دارای دو دستور باشد، دستورات به ترتیب اجرا می‌شود. فاکشن لوپ IOOP یکی از اساسی ترین بخش های برنامه نویسی در آردوینو است و بدون آن تداخل در عملکرد میکروکنترلر ایجاد می‌کند. حتی اگر در این قسمت برنامه ای هم نوشته نشده باشد، نیاز به آن خواهد بود.

```
١ } ( ) loop void
٢ : put your main code here, to run repeatedly //
٣
٤ {
```

## Custom functions – فانکشن های سفارشی

دو بخش اصلی که تا به حال به آن پرداخته شده است، فقط از دو تابع اصلی است. در برنامه های طولانی و پیچیده که شامل چندین بخش منطقی و مستقل از هم هستند، بهتر است برای هر قسمت منطقی برنامه ای جداگانه نوشته شود. برای نوشتن تابع باید اهداف تابع مشخص باشد. چه وظیفه ای بر عهده دارد، ورودی و خروجی ها مشخص باشد. بخش های اصلی برنامه به صورت زیر است.

- a return type
- a name
- a list of parameters

نوع تابع یکی از روش های برنامه نویسی C++ است که توسط ما تعریف می‌شود. اگر تابعی بخواهد مقداری را به تابع فراخوان برگرداند، آن مقدار در نام تابع قرار می‌گیرد. هر مقدار دارای نوع است و نام تابع هم باید دارای نوع باشد. اگر تابع هیچ مقداری را به برنامه فراخوان برنگرداند، نوع آن VOID خواهد بود.

```

۱ < نوع تابع > نام تابع ( لیست پارامترها )
۲
۳ }
۴ <دستور اول>
۵ <دستور دوم>
۶
۷ .
۸ .
۹ .
۱۰ دستور n
۱۱ }

```

- اگر تعداد پارامترها بیش از یکی باشد، باید با کاما , از یکدیگر جدا شود.
- برای اجرای توابع آن ها را با نامشان فراخوانی کنید.
- متغیرهای مورد نیاز توابع را در داخل توابع تعریف کنید. هیچ تابعی نمیتواند از متغیرهای توابع دیگر استفاده کند.
- تعریف تابع در داخل تابع دیگر امکان پذیر نیست.

## تست برنامه با void setup

یک برنامه بسیار ساده برای روش تست voidsetp در ادامه نوشته ایم. در ابتدا برنامه ی آردوینو را باز کنید.

```

۱ } () setup void
۲ :put your setup code here, to run once //
۳ ; (۹۶۰۰) begin . Serial
۴ ; (" !Hello World") println . Serial
۵ {
۶
۷ } () loop void
۸ :put your main code here, to run repeatedly //
۹
۱۰ }

```

در بخش voidsetup توابع اصلی که فقط یک بار در برنامه اجرا می شود را مینویسیم. در این برنامه از سریال مانیتور که برای نمایش دیتا در نرم افزار آردوینو است، استفاده می کنیم.

- از سریال مانیتور برای ارتباط بین دو آردوینو با یکدیگر، آردوینو به کامپیوتر و یا ارتباط آردوینو با سایر دستگاه ها استفاده می شود.

- آردوینو یک پورت سریال دارد که با نام UART یا Universal Asynchronous Transceiver Receiver (UART) مشخص شده است که به فرآیند ارسال داده با n بیت اطلاعات در یک لحظه گفته می‌شود.
- در آردوینو از پین های دیجیتال ۰ و ۱ برای راه اندازی پورت سریال استفاده می‌شود.

## سریال مانیتور – Serial Monitor

نواع سریال مانیتور

• تابع `Begin`

• توسط تابع `begin` برقراری ارتباط سریال و سرعت انتقال دیتا مشخص می‌شود.

`Serial.begin (speed);`

• تابع `print`

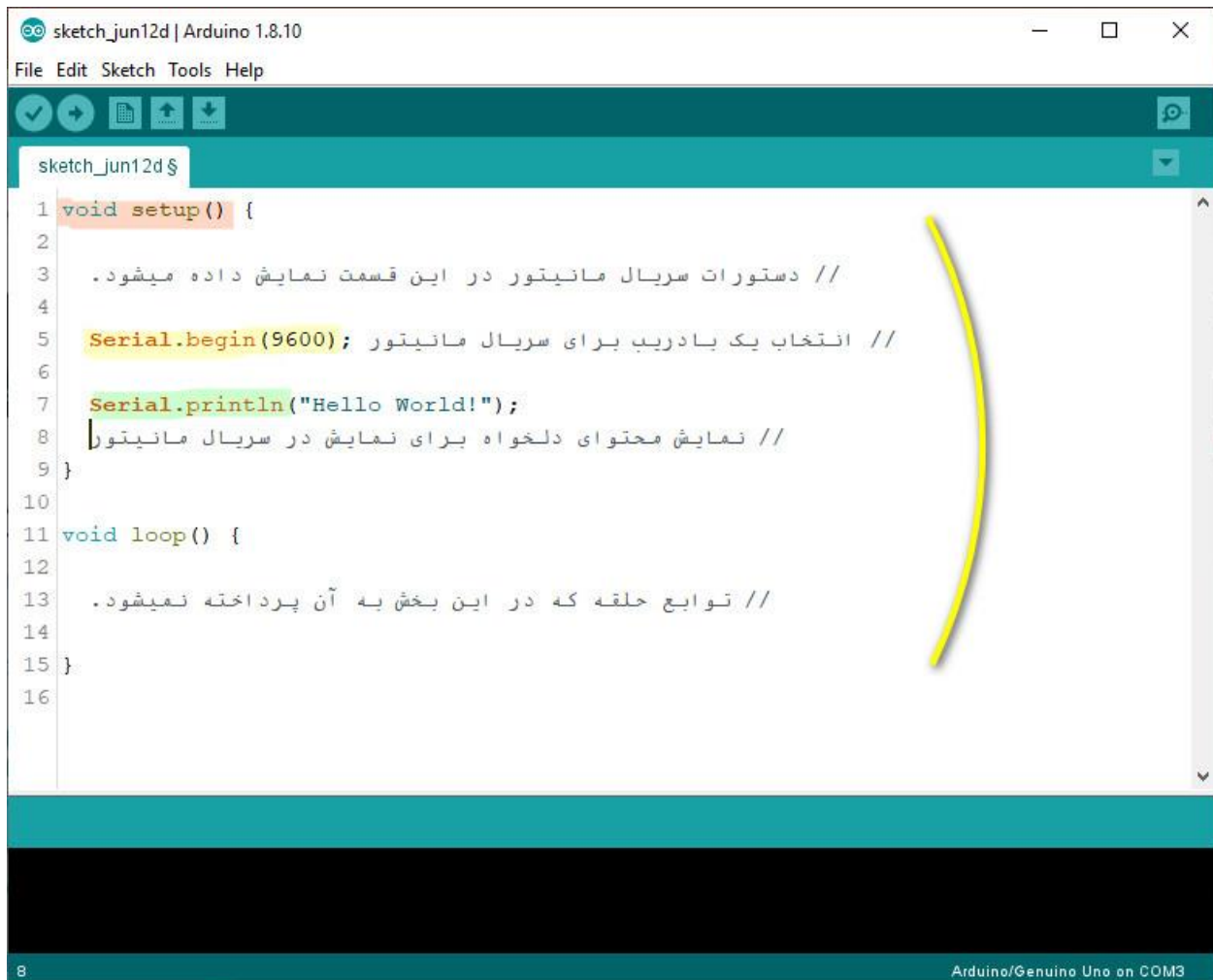
• دیتا را به پورت سریال ارسال می‌کند.

`Serial.print (78);`

• تابع `println`

• همانند تابع `print` دیتا را به پورت سریال ارسال می‌کند با این تفاوت که با هر بار ارسال اطلاعات دستور رفتن به خط بعد را اجرا می‌کند.

`Serial.println(78);`



```
sketch_jun12d | Arduino 1.8.10
File Edit Sketch Tools Help
sketch_jun12d $
1 void setup() {
2
3 // دستورات سریال مانیتور در این قسمت نمایش داده میشود.
4
5 Serial.begin(9600); // انتخاب یک بادریب برای سریال مانیتور
6
7 Serial.println("Hello World!");
8 // نمایش محتوای دلخواه برای نمایش در سریال مانیتور
9 }
10
11 void loop() {
12
13 // توابع حلقه که در این بخش به آن پرداخته نمیشود.
14
15 }
16
```

8 Arduino/Genuino Uno on COM3

برای ورود به سریال مانیتور CTRL + SHIFT + M را همزمان نگه دارید و یا بر روی آیکون آن در سمت راست نرم افزار کلیک کنید.



## تست برنامه با VOIDLOOP

همان کد قبلی را در حلقه ی برنامه تعمیم میدهیم.

```
1 } () setup void
2 // توابع اصلی که فقط یک بار اجرا میشود.
3 ; (۹۶۰۰) begin . Serial
4 ; ("HELLO WORLD") println . Serial
5 {
6
7 } () loop void
8 // توابعی که مداوم تا پایان حلقه در برنامه اجرا میشود.
9 ; ("! GOOD LOCK") println . Serial
10 ; (۱۰۰۰) delay
11 }
```

دستورات سریال مانیتور دقیقاً مشابه قبل است.



```
sketch_jun12d | Arduino 1.8.10
File Edit Sketch Tools Help
sketch_jun12d $
1 void setup() {
2 // توابع اصلی که فقط یک بار اجرا میشود.
3 Serial.begin(9600);
4 Serial.println("HELLO WORLD");
5 }
6
7 void loop() {
8 // توابعی که مداوم تا پایان حلقه در برنامه اجرا میشود.
9 Serial.println("GOOD LOCK!");
10 delay(1000);
11 }
12 |
Invalid library found in C:\Users\Lenovo\Documents\Arduino\libr
Invalid library found in C:\Users\Lenovo\Documents\Arduino\libr
12 Arduino/Genuino Uno on COM3
```

در این بخش برای تکرار دستور مورد نظر از تابع زمانی DELAY استفاده می‌کنیم.

## Delay()

از تابع Delay برای متوقف کردن برنامه برای چند میلی ثانیه استفاده می‌شود.

**delay(ms)**

مقداری بر حسب میلی ثانیه برای متوقف کردن برنامه = ms

برای ورود به سریال مانیتور CTRL + SHIFT + M را همزمان نگه دارید و یا بر روی آیکون آن در سمت راست نرم افزار کلیک کنید.

