СИНТАКСИС

* Тела функций заключаются в фигурные скобки  { } . Код внутри фигурных скобок иногда называют **блоком кода**.
* Каждая команда заканчивается точкой с запятой ;
* Метод применяется к объекту через точку. Пример: Serial.begin();
* Вызов функции или метода всегда заканчивается скобками, даже если функция не принимает параметров. Пример: loop()
* Разделитель десятичных дробей - **точка**. Пример: 0.25

 У запятой тут другое применение.

* Запятыми перечисляются аргументы функций и методов, члены массива, также через запятую можно выполнить несколько действий в одну строчку. Пример: digitalWrite(3, HIGH); массив - int myArray[] = {3, 4, 5 ,6};
* Одиночный символ заключается в одиночные кавычки 'а'
* Строка и массив символов заключается в двойные кавычки "строка"
* Имена переменных могут содержать **латинские буквы** в верхнем и нижнем регистре (большие и маленькие), **цифры** и **подчеркивание**. Пример: myVal\_35 .
* Имена переменных **не могут начинаться с цифры**. Только с буквы или подчёркивания.
* **Регистр имеет значение**, т.е. большая буква отличается от маленькой. Пример: имена val и Val - не одно и то же.

К синтаксису так же относят *комментарии*. В каждом языке комментарии выделяются по-разному, но едино то, что комментарии, по сути, не являются частью кода. Комментарии нужны для пояснения кода себе или пользователю:

// Просто комментарий

ОФОРМЛЕНИЕ

Есть такое понятие, как форматирование или выравнивание кода, то есть соблюдение пробелов и интервалов. В профессиональных средах разработки зачастую есть авто-форматирование кода. Arduino IDE – тоже имеет такое, в нём код форматируется комбинацией клавиш **Ctrl + T**.

* Между математическими действиями, знаками сравнения, присваивания и всем подобным ставится **пробел**.
* Как и в обычном тексте, пробел **ставится после** и **не ставится перед** запятой, двоеточием, точкой с запятой.
* Отступ от левого края экрана - знак табуляции, код сдвигается вправо и на одном расстоянии формируются команды из одного блока кода. В Arduino IDE одна табуляция равна двум пробелам. Можно использовать клавишу **Tab**.
* Каждое действие выполняется с новой строки (авто-форматирование это не исправляет).
* Имена функций и переменных принято называть с маленькой буквы. Пример: value
* Если имя состоит из двух и более слов, то их принято разделять. Есть два способа:
  + **camelCase** (верблюжий стиль): первая буква маленькая, каждая первая буква следующего слова - большая.
  + **under\_score** (подчёркивание): все буквы маленькие, разделитель - подчёркивание.
* Имена типов данных и классов принято писать с большой буквы. Пример: Signal, Servo
* Имена констант принято писать в верхнем регистре, разделение - подчеркивание. Пример: MOTOR\_SPEED
* При написании библиотек и классов, имена внутренних (приватных) переменных принято писать, начиная со знака подчёркивания. Пример: \_position
* Несколько общепринятых сокращений для названий переменных, вы часто будете встречать их в чужих прошивках и библиотеках:
  + **button** - btn, кнопка
  + **index** - idx - i, индекс
  + **buffer** - buf, буфер
  + **value** - val, значение
  + **variable** - var, переменная
  + **pointer** - ptr, указатель
* Имена функций и методов принято начинать с глагола, кратко описывающего действие функции. Вот те из них, которые вы будете встречать постоянно:
  + **get** - получить значение (getValue)
  + **set** - установить значение (setTime)
  + **print**, **show** - показать что-то
  + **read** - прочитать
  + **write** - записать
  + **change** - изменить
  + **clear** - очистить
  + **begin**, **start** - начать
  + **end**, **stop** - закончить, остановить

СТРУКТУРА КОДА

Переменная любого типа должна вызываться только **после** своего объявления. Иначе будет ошибка. К тому же, объявление и использование классов или типов данных из библиотеки/файла должно быть **после** подключения библиотеки/файла

При запуске Arduino IDE даёт нам заготовку в виде двух обязательных функций: **setup()** и **loop()**.

Код в блоке setup() выполняется один раз при каждом запуске скетча. Код в блоке loop() выполняется постоянно (повторяется) на всём протяжении работы программы, начиная с момента завершения выполнения блока setup().

БИБЛИОТЕКИ

 Библиотеки подключаются в главный файл (файлу прошивки) при помощи директивы #include. Данная команда сообщает компилятору, что нужно найти и добавить в программу указанный файл. Этот файл может содержать свои #include и тянуть за собой и другие файлы, таким образом программа может быть разбита на множество независимых файлов.

#include <Servo.h> // подключает библиотеку Servo.h

#include “Servo.h” // тоже подключает библиотеку Servo.h

Когда указываем название "в кавычках", компилятор сначала ищет файл в папке со скетчем, а затем в папке с библиотеками. При использовании <галочек> компилятор ищет файл только в папке с библиотеками.

МЫШКА БЫВАЕТ НЕ НУЖНА

Чем меньше вы будете использовать компьютерную мышь для установки курсора в нужное место или выделения слов и строк, тем меньше времени вы потратите в общем.

* **Ctrl+←**,**Ctrl+→**– переместить курсор влево/вправо НА ОДНО СЛОВО
* **Home** , **End** – переместить курсор в начало/конец строки
* **Shift+←**,**Shift+→**– выделить символ слева/справа от курсора
* **Shift+Ctrl+←**,**Shift+Ctrl+→**– выделить слово слева/справа от курсора
* **Shift+Home**,**Shift+End**– выделить все символы от текущего положения курсора до начала/конца строки
* **Ctrl+Z** – отменить последнее действие
* **Ctrl+Y** – повторить отменённое действие
* **Ctrl+C** – копировать выделенный текст
* **Ctrl+X** – вырезать выделенный текст
* **Ctrl+V** – вставить текст из буфера обмена

Местные сочетания(для среды программирования):

* **Ctrl+U** – загрузить прошивку в Arduino
* **Ctrl+R** – скомпилировать (проверить)
* **Ctrl+Shift+M** – открыть монитор порта
* **Ctrl+T** – авто-форматирование

Для отодвигания комментариев в правую часть кода используйте TAB, а не ПРОБЕЛ. Нажатие TAB перемещает ваши комментарии на одинаковом расстоянии (словно по строке).