**Синтаксис и структура кода**

**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.
* Однострочный комментарий

// однострочный комментарий

* Многострочный комментарий

/\* Многострочный

комментарий \*/

**Оформление**

Во всех серьезных средах разработки есть автоформатирование кода, оно работает как в процессе написания, так и "по кнопке". Arduino IDE - не исключение, в ней код форматируется комбинацией клавиш **Ctrl+T**:

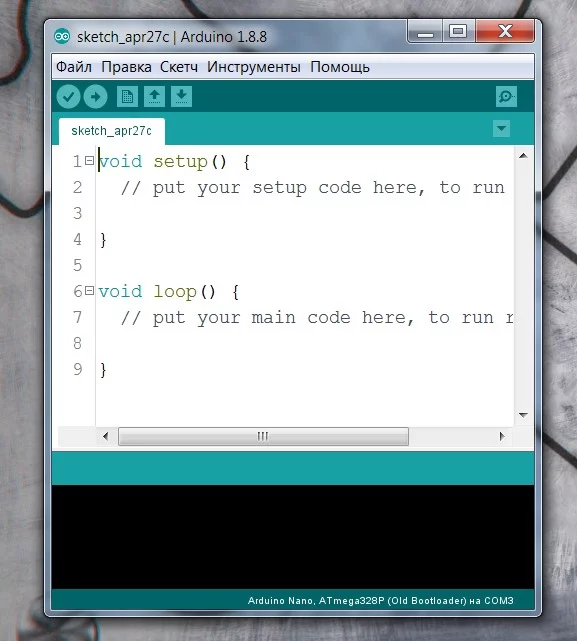
* Между математическими действиями, знаками сравнения, присваивания и всем подобным ставится **пробел**.
* Как и в обычном тексте, пробел **ставится после** и **не ставится перед** запятой, двоеточием, точкой с запятой.
* Отступ от левого края экрана - знак табуляции, код сдвигается вправо и на одном 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 (автоформатирование 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**

Прежде чем переходить к структуре и порядку частей кода, нужно кое-что запомнить:

* Переменная любого типа должна вызываться только **после** своего объявления. Иначе будет ошибка
* Объявление и использование классов или типов данных из библиотеки/файла должно быть **после** подключения библиотеки/файла
* Функция может вызываться **как до, так и после** объявления, потому что C++ компилируемый язык, компиляция проходит в несколько этапов, и функции "выделяются" отдельно, поэтому могут вызываться в любом месте программы

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 МК. Код в блоке loop() выполняется "по кругу" на всём протяжении работы программы, начиная с момента завершения выполнения setup().

**Подключение библиотек и файлов**



* Пользовательская папка: *Документы/Arduino/libraries.* Сюда библиотеки попадают при добавлении их через "подключить .zip библиотеку" и при установке из менеджера библиотек.
* Папка с программой: *C:/ProgramFiles(x86)/Arduino/libraries* (или *C:/Program Files/Arduino/libraries* для 32-разрядной Windows). Здесь хранятся встроенные стандартные библиотеки.