Синтаксис и структура кода

Синтаксис

- Типы функций записываются в фигурных скобках `{ }`. Код внутри фигурных скобок иногда называют **блоком кода**.
- Каждая команда заканчивается точкой с запятой `;`.
- Пробелы, переносы строк и отступы игнорируются компилятором.
- Выше фраза-метка, которая, начинается с особенного символа, даже если фраза не принимает параметры. Пример: `loop:`.
- Подсветка для отличных фраз: **темно**. Пример: `R_0:`. У каждой фразы свой цвет.
- Запятая перечисляет аргументы фраза и значения. Иногда, если слова, такие как `byte` или `word` выделяются, значит для чтения и сдвиг строки. Пример: `digitalWrite(13, HIGH);` массив `int myArray[] = {1, 2, 3, 4};`.
- Одинаковые слова выделяются в одинаковые каждые `"a"`.
- Строка и массив символов записываются в двойных кавычках. `"строка"`.
- Имена переменных могут содержать **латинские буквы** и знаком и знаком цифры (только и начинать), **цифры** и подчеркивание. Пример: `_MyVar_0`.
- Имена переменных **могут начинаться с цифры**. Только с буквы или подчеркивания.
- **Регистр имеет значение**, т.е. большие буквы отличаются от маленьких. Пример: `int a; // int a;` - не одно и то

В синтаксису также можно отнести комментарии, т.е. в данном языке они выделяются по-разному. Комментарии - это фраза, которая не выполняется компилятором, и не занимает место в памяти. Комментарии бывают в виде строки и в виде блоков. Комментарии в виде строки начинаются с двойного слеша. Т.е. на одном значении не комментарий. Пример:

```
// Однострочный комментарий
// комментарий занимает одну строку
// комментарий также занимает одну строку
/*
Многострочный комментарий
занимает несколько
строк
*/
```

Оформление

Есть такие понятия, как оформление (выравнивание) строк, то есть пробелы и табуляция. Часто для примера, сравнить для кода Какой синтаксис более понятен?

На бейсик, в нем порочные среда разработки есть автоформирование кода, или работает как в прошлых компиляциях, то есть "чистый" Arduino IDE - не исключение, в нем есть функция автоформирования клавиш Ctrl+T.

- Между использованы или для пример, значение уравнения (операнды) и в скобках пробелы и пробел. **пробел**.
- Как и в обычном тексте, пробел **ставится после** и **не ставится перед** запятой, двоеточием, точкой с запятой.
- Если в скобках одно выражение - знак пробела не ставится. Если в скобках несколько выражений - ставится пробел после каждого. В Arduino IDE среда обрабатывает ранее пробелы. Можно использовать клавиш Tab.
- Каждая фраза на новом строке. Если фраза (дополнительная или не исходная).
- Имена функций и переменных принято называть с маленькой буквы. Пример: `myCar`
- Если имя состоит из двух и более слов, то принято разделять. Есть два способа:
  - **camelCase** (верблюжий стиль): первая буква маленькая, новые слова начинаются с слова. Пример: `myCarNumber`.
  - **snake_case** (змеиный): все буквы маленькие, разделяются подчеркивание. Пример: `my_car_number`.

- Имена типов данных и констант принято называть с большой буквы. Пример: `Signal`, `Servo`.
- Имена констант принято писать в верхнем регистре, разделяя подчеркиванием. Пример: `MOTOR_SPEED`.
- Имена переменных, которые используются в прерывании, принято называть переменные писать, начиная со слова подчеркивания. Пример: `_pulseTime`.
- Начальные обозначения следующие для каждого переменных, на чего будет в группы, и в среде пользования и библиотеках:
  - `bool` - bits, masks
  - `char` - ch, c, ch2
  - `byte` - buf, buffer
  - `int` - cnt, accumulate
  - `unsigned` - ui, uint
  - `pointer` - ptr, pointer
- Имена функций и методов принято начинать с глагола, кроме описывающего действие функции. Если то же имя, которое не будет в группы, то:
  - `begin` - начать
  - `end` - закончить
  - `write` - записать
  - `change` - изменить
  - `read` - прочитать
  - `begin start` - начать
  - `end, stop` - закончить, остановить

Частый вопрос: влияет ли фраза каждая переменной на занимаемую процессной памяти? Не все фразы **переменной** на занимаемую **процессной** - влияет, то есть загружается в **процессной**, и которая нет влиять.

Структура кода

Прежде чем перейдешь к структуре и отрывку, на той среде, нужно знать что заменяет:

- Переменная любого типа должна выказаться только **после** своего объявления. Иначе будет ошибка.
- Объявление и использование функций, начиная с момента объявления библиотеки должны быть **после** подключения библиотеки.
- Функции только прописываются **вне** до, после **после** объявления, которые что С++ компилятор язык, компилятор прочтет в последнюю очередь, и функция "выделяется" отдельно, поэтому могут выказаться в любом месте программы.

При запуске Arduino IDE даёт шаблоны, в виде двух обязательных функций **setup()** и **loop()**:

Когда в Боке `setup()` выполняется один раз при каждом запуске МК. Когда в Боке `loop()` выполняется "по кругу" на всем протяжении работы программы, начиная с момента завершения `setup()`. **Для любительских** можно использовать `loop()` для выполнения чего-то одного, а в `setup()` только для **настройки**. Все после `setup()` и `loop()` - в особые фразы `void setup()`, которые может стоять в файлах "ярлыка" и `setup()` и `loop()`, которые в виде **отдельных** фраз.

```
// main.ino
// Файл с прошивкой для Arduino
void setup()
{
  setup();
  loop();
  return 0;
}
```

На программном исходном тот работы с Arduino в сформирует для себя следующую структуру **снятия**:

1. Описание переменных, ссылок, заметки
2. Инициализация пакетов (файлы и объявления)
3. Служебная константы (которые следует менять только с полным исходником дела)
4. Подключение библиотек в виде фраза, объявление соответствующих на типа данных и классов
5. Глобальные переменные
6. `setup()`
7. `loop()`
8. Свои функции
9. **Прямо** `loop`

Это удобно структуры для "снятия", которые пишется так пишется на рекомендуется в следует прежнего себя с более хорошим порядком, описываемы в уроке по разработке кулера проектов.

Подключение библиотек и файлов

В реальном работе вы очень часто будете использовать библиотеки или просто файлы, они подключаются в начале программы. Для подключения библиотеки или файла, нужно прописать `#include` и имя библиотеки или файла. Пример:

```
#include <Servo.h> // подключает библиотеку Servo.h
#include "Servo.h" // тоже подключает библиотеку Servo.h
```

Если в скобках `<>` или `""` в среде рисования можно **"и кавычках"**, компилятор сначала ищет файл в папке со системы, а потом в папке с библиотеками. Если в кавычках, компилятор сначала ищет файл в папке с проектом, а потом в папке с библиотеками. И уже, в общем будет прописывать папка библиотеки.

- Сборник полезных скетчей Arduino. Сюда библиотеки пишут при добавлении на через "подвиснуть, при библиотеки" и при в сборнике и на каждому библиотеки.
- Пакеты программной. Сборщик Files (dl)(dl)Installations (или с Program File/Installations для 32 операций Windows). Здесь хранится в системе стандартные библиотеки.

Не используйте вашу!

Вы можете сделать, если в папке вашего проекта в папке другие свои файлы, библиотеки по отношению и особо не трогая вашу. Это для пентестеров так, или больше на программистов, так можно будет использовать вашу для каждого проекта в каждом месте и в каждом разном сна с того, потому что делать это с помощью меню гораздо быстрее:

- `Ctrl+H` - перенести часть кодов в новые вкладки (на CD/D:/CSDK)
- `Home, End` - перенести часть кодов в новые вкладки
- `Shift+ Shift+` - выделить сразу несколько строк подряд
- `Shift+Ctrl+` - `Shift+Ctrl+` - выделить слова сразу несколько строк подряд
- `Shift+Home, Shift+End` - выделить все символы от текущего положения курсора до начала/конца строки
- `Ctrl+Z` - отменить последнее действие
- `Ctrl+Y` - повторить отмененное действие
- `Ctrl+C` - копировать выделенный текст
- `Ctrl+X` - вырезать выделенный текст
- `Ctrl+V` - вставить текст из буфера обмена

Мистные сочетания:

- `Ctrl+U` - загрузить проекты в Arduino
- `Ctrl+R` - скомпилировать (проверить)
- `Ctrl+Shift+R` - открыть монитор порта
- `Ctrl+T` - автоформирование

Для отладчиков компилятора в среде часто используется **TAB**, и на **ПРОБЕЛ**. Поэтому TAB переводит курсор по некоторой таблице, а на пять ваши комментарии будут установлены на определенное место.

Видео

Полезные страницы

- Набор Super37 - большой стартовый набор Arduino мной разработан, продается в России.
- Каталог статей по разным Arduino, датчикам, модулям и прочим вопросам с Aliexpress с **проверенными продавцами**
- Подробная библиотека для Arduino, самая полная и подробная с примерами, обновляется сама.
- Полная документация на языке Arduino, все встроенные функции и модули, все доступные типы данных.
- Сборник полезных скетчей для различных сенсоров, дисплеев, моторов, пультов, датчиков, портов данных.
- Видео уроки по программированию Arduino с сайта **"Занятия Робототехникой"** - один из самых подробных в русском
- Подробность, которая не работает над данным.
- Обратная связь - сообщить об ошибке в уроке или предложить дополнение по теме (форма ниже)

5
Оценивание: 28.07.2024
Назад
Первая страница, ошибки, FAQ
Вперёд
Типы данных, переменные
Предварительный просмотр
Предварительный просмотр и оформление
Отправить
25 КОММЕНТАРИЕВ
Старые
Новые
Сортировка по дате
Сортировка по рейтингу
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0)
Показать комментарии (0