Привет ребята, в этом выпуске уроков ардуино мы будем изучать циклы. Умение работать с циклами позволит вам писать компактные и очень эффективные коды.

Начнем с того что с циклами вы уже по-любому сталкивались. Это конечно же основной цикл любой программы луп. Цикл это грубо говоря рамки, код внутри которых выполняется сверху вниз и повторяется с начала, когда достигает конца, и продолжается это дело до тех пор, пока выполняется какое то условие. Есть два основных цикла, с которыми мы будем работать, это фор и вайл.

Цикл фор, в простонародии счётчик, на вход ему подается три значения: во первых переменная счетчика, принято для этого создавать новую и начинать цикл с нуля, но можно использовать просто любую переменную.

Далее идет условие, всё разделяется точкой запятой. Пока это условие верно, цикл будет выполняться. Условие может быть например пока наша переменная меньше 100. Если условие нарушится, то программа пойдет дальше, минуя весь блок. Третьим идет изменение переменной, например с каждом шагом цикла будем увеличивать переменную на 5. Можно увеличить на 1, используя стандартную штуку ++. Или уменьшить на 1, используя --. Давайте предскажем работу цикла. Первая итерация: счётчик, переменная и равна 0, условие выполнено, весь код заключенный в фигурных скобках выполняется. Далее следует возврат в начало цикла. И увеличивается на 1, снова проверяется условие. Верно? Да, код выполняется. И так далее. Как только достигаем значения 100, условие не выполняется и мы выходим из цикла.

Самое главное здесь, что переменную и мы можем использовать, получая ее значение. Давайте выводить ее значение в порт при каждой итерации цикла. Откроем порт, и сделаем вывод каждый раз с новой строки. И поставим задержку. И после цикла поставим задержку, сейчас всё увидите.

Вот так вот, значение доходит до 99, и цикл запускается заново, так как он сам находится в цикле луп, цикл в цикле, а вы как хотели. Есть очень важная команда, брейк. Она позволяет выйти из цикла в любой нужный момент. Вот вам пример: внутри цикла создадим условие, если и больше 50, то выйти из цикла. Как видите, мы вышли из цикла в нужном месте. Это используется для каких то специфических целей, нужно уметь пользоваться. Ещё есть оператор континю, продолжить. Лично я его ещё ниразу не использовал, но работает он как пропусти ход. То есть цикл сразу идёт в начало, пропуская все что после оператора континуе. Знать надо, может где и пригодится.

Цикл фор используется для самых различных целей. Например для работы с массивами, но про массивы будет отдельный урок.

Для массового управления несколькими пинами. Например 6 светодиодов в моей маске для осознанных сновидений. Порядок вспыхивания светодиодов задается именно в циклах.

Даже сформировать кучу пинов как выходы можно при помощи цикла. Вот так вот это будет выглядеть. Надеюсь это вам понятно, ничего сложного здесь нет. И давайте для примера зажгем и погасим линейку светодиодов. Первый цикл будет включать светодиоды по очереди. А второй выключить. Вы уже представили, что без циклов данный код будет выглядеть вот так.

Второй цикл который мы сегодня рассмотрим, работает чуть чуть иначе, это цикл вайл. В скобках пишется условие, то есть выражение читается так: пока выполняется условие, делать то что в фигурных скобках. Как только условие перестанет выполняться - цикл будет пропущен.

Циклом вайл можно например принудительно ожидать ввода значений в монитор порта. То есть пока значение не получим, код дальше не пойдет.

Из вала очень просто можно сделать цикл фор, это вам чисто для развития. Видите, суть то одна и та же.

На цикле вайл можно сделать такую штуку: в условие заранее ставим единицу, то есть условие всегда верно. Такой цикл будет выполняться бесконечно, программа на нем грубо говоря зависнет. Чтобы выйти, используем условие и брейк. Например так можно организовать принудительный опрос датчика, программа не пойдет дальше пока не получит нужное значение. Иногда я использую такую конструкцию, когда не хочу делать кучу условий для прозрачности кода. Например в проекте с имитатором бомбы из контр страйк я таким образом ждал ввода паролей. Это костыль, это не эффективный код, это плохо, хорошие программисты так не делают. Но в некоторых случаях так делать можно, это упрощает код и делает его понятнее.

У цикла вайл есть разновиднось, можно сказать его противоположность, отличается только местом где задается условие, это цикл Ду. Цикл вайл не выполнится, если условие неверно. А вот цикл ду выполнится хотя бы один раз, даже если условие неверно. Так как условие задается в конце цикла. То есть это можно прочитать так: делать набор функций, пока условие верно. Лично я им ещё ниразу не пользовался, но знать о такой штуке надо.

Вот и все, теперь мы знаем про циклы и будем использовать их в дальнейших уроках. Не забудьте подписаться если ещё не подписаны, до встречи!