Эксперимент 1. Привет, Мир!

После установки среды программирования, драйвера и подключения Лаборатории IoT к компьютеру, наконец, можно приступить к первому эксперименту.

Когда программисты впервые знакомятся с новым языком программирования, то первым делом они пишут программу «Hello, World!» (Привет, Мир!). Все, что она делает— это выводит эту приветственную надпись на экран. Никакой полезной функции такая программа, казалось бы, не несет, но это заблуждение. Главная задача этой программы— убедиться, что все настроено верно, и что всё работает. Только после этого можно двигаться дальше, к более сложным программам.

Приступим

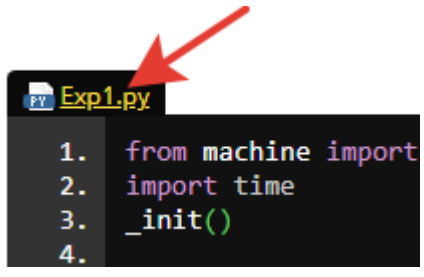
Классическим «Hello, World!» в мире программирования микроконтроллеров является мигание светодиодом. Именно это и будет наш первый эксперимент. Ниже представлен листинг программы первого эксперимента:

Exp1.py

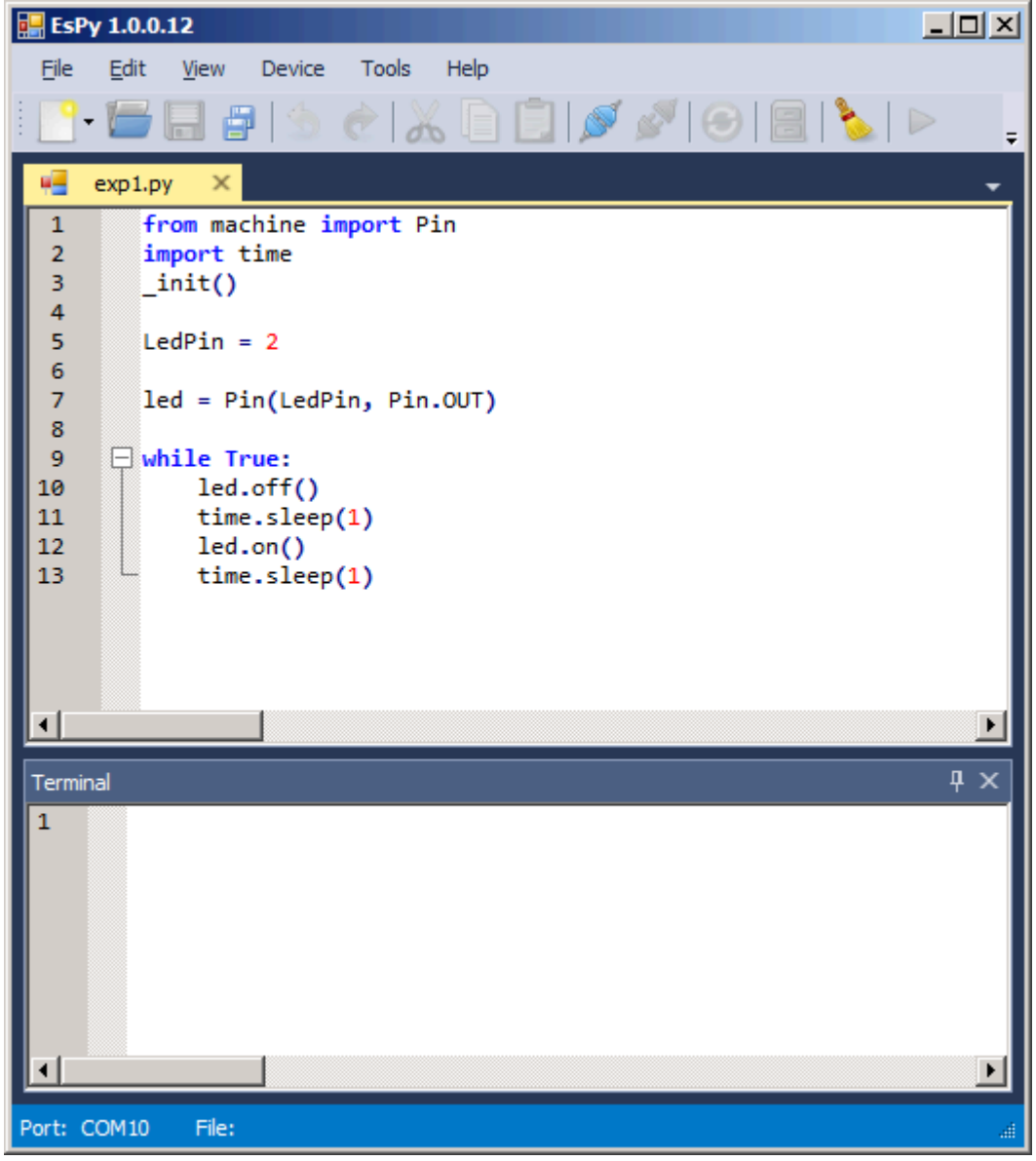
```
1.  from machine import Pin
2.  import time
3.  _init()
4.
5.  LedPin = 2
6.
7.  led = Pin(LedPin, Pin.OUT)
8.
9.  while True:
10.     led.off()
11.     time.sleep(1)
12.     led.on()
13.     time.sleep(1)
```

Введи код первой программы в среду программирования EsPy. Это можно сделать тремя способами:

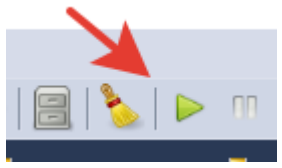
1. Набрать код программы в EsPy вручную
2. Скопировать код и вставить его в EsPy
3. Скачать код в виде файла и открыть его в среде EsPy. Чтобы скачать файл нужно нажать на его название над листингом (показано на рисунке красной стрелкой)



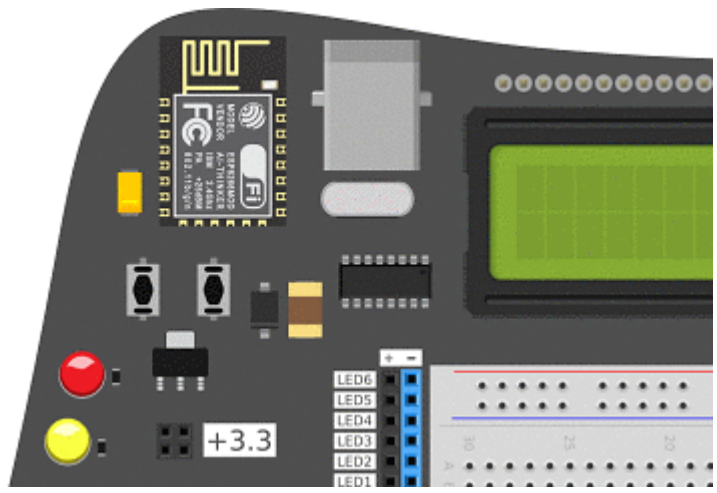
Так или иначе код программы должен оказаться в EsPy



Необходимо убедиться, что EsPy установлено соединение с Лабораторией IoT (Подробнее о том, как это сделать) и отправить листинг интерпретатору Python, который находится в микроконтроллере.



Результатом должно стать мигание синего светодиода в верхнем левом углу Лаборатории.



Не работает?

Если не получилось, то необходимо еще раз проверить подключение, убедиться, что соединение настроено верно. Для этого еще раз внимательно прочитать статью Подключение и настройка. Если все выполнено верно, но все равно не работает — Возможные проблемы и их устранение

Как это работает?

Теперь, когда первый эксперимент удался, самое время разобраться как и почему это работает.

В первых двух строках, с помощью оператора `import` мы подключаем необходимые системные библиотеки, а именно `time` и класс `Pin` из библиотеки `machine`

```
1. from machine import Pin
2. import time
```

`time` отвечает за измерение времени, функции даты и за задержки, а `Pin` реализует возможность обращаться к портам GPIO (*англ. general-purpose input/output* по-русски *Интерфейс ввода/вывода общего назначения*), с его помощью можно настроить вывод микроконтроллера и подать на него напряжение.

> **Подробнее об import и from [https://pythonworld.ru/osnovy/rabota-s-modulyami-sozdanie-podklyuchenie-instrukciyami-import-i-from.html]**

Вызываем функцию `_init()` — это функция, которую мы написали и заботливо положили в память Лаборатории IoT при производстве. Она необходима для сброса состояния микроконтроллера в начальное состояние, чтобы результаты запуска другого эксперимента не влияли на текущий.

```
3.  _init()
```

В строке 5 мы объявляем переменную `LedPin` и присваиваем ей значение `2`

```
5.  LedPin = 2
```

Как видно из её названия, она хранит номер вывода микроконтроллера, к которому подключен светодиод. Все выводы микроконтроллера имеют свой номер, именно по этому номеру и происходит работа с ними из программы.

Далее нам нужно настроить вывод микроконтроллера:

```
7.  led = Pin(LedPin, Pin.OUT)
```

Мы вызываем функцию Pin и передаем ей в качестве первого параметра номер вывода (который записан в переменной `LedPin`), а вторым параметром мы сообщаем, что хотим настроить этот вывод как цифровой *Выход*. Результатом функции `Pin` является одноименный объект `Pin` вывода микроконтроллера, который мы записываем в переменную `led` .

Теперь мы можем обращаться к этому объекту и влиять на состояние ножки микроконтроллера.

```
 9.  while True:
10.      led.off()
11.      time.sleep(1)
12.      led.on()
13.      time.sleep(1)
```

Функция `led.off()` устанавливает низкий уровень на выводе `led` , а функция `led.on()` — высокий. Таким образом мы подаем или снимаем напряжение со светодиода, который подключен к этому выводу.

Зачем же нам `time.sleep(1)` ? Все дело в том, что микроконтроллер работает очень быстро, исполняя миллионы операций в секунду. Он может тысячи раз в секунду зажечь и потушить светодиод. Это слишком быстро, чтобы человек смог что либо заметить. Чтобы приостановить исполнение программы мы и вызываем функцию `time.sleep` . В качестве параметра она принимает число секунд на которое нужно сделать паузу. В нашем случае мы делаем паузу на 1 секунду после зажигания светодиода и столько же после его выключения.

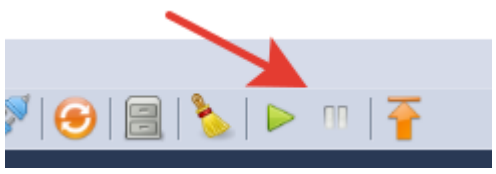
Но что такое `while True:` ? Это цикл. Циклы— это важнейшие элементы программ. Они нужны, чтобы выполнять один и тот же участок кода несколько раз. В языке Python есть несколько видов циклов и `while` это один из них. Этот цикл выполняется до тех пор, пока выполняется условие, пока оно *истинно*. В нашем случае это константа `True` , что переводится как *истина*. Получается, что наше условие всегда безусловно *истинно*, и наш цикл является бесконечным. Участок кода, который зажигает и гасит светодиод будет продолжаться бесконечно, пока на плату поступает питание и, пока, не будет прервано выполнение программы извне.

Как Python понимает какой именно участок кода является *телом цикла*, какой именно участок кода необходимо повторять? Для этого в Python используются отступы. Как можно видеть код в строках с 10 по 13 начинается не с начала, а с некоторым отступом. Вот именно тот участок кода, который обладает данным отступом и есть *тело цикла*.

- **Подробнее о циклах в Python [https://pythonworld.ru/osnovy/cikly-for-i-while-operatory-break-i-continue-volshebnoe-slovo-else.html]**
- **Подробнее о синтаксисе языка Python [https://pythonworld.ru/osnovy/sintaksis-yazyka-python.html]**

Остановка программы

Как же прервать исполнение программы, когда она занята исполнением бесконечного цикла, ведь сама она никогда не закончится? Нажмите на клавиатуре `Ctrl+I` или на кнопку



В окне терминала появится сообщение `KeyboardInterrupt:` (Прерывание от клавиатуры)