- **asyncio module**
  - **Module Functions**
    - asyncio.run(coro, *, debug=False)
    - coroutine asyncio.sleep(delay, result=None)
    - **Introspect**
      - asyncio.current_task(loop=None)
      - asyncio.all_tasks(loop=None)
      - asyncio.iscoroutine(obj)
      - asyncio.iscoroutinefunction(func)
    - **Many Tasks**
      - awaitable asyncio.gather(*aws, return_exceptions=False)
      - coroutine asyncio.wait(...)
        - asyncio.ALL_COMPLETED
        - asyncio.FIRST_COMPLETED
        - asyncio.FIRST_EXCEPTION
      - asyncio.as_completed(aws, *, timeout=None)
    - **Tasks**
      - asyncio.create_task(coro, *, name=None)
      - awaitable asyncio.shield(aw)
      - coroutine asyncio.wait_for(aw, timeout)
    - **Thread**
      - coroutine asyncio.to_thread(func, /, *args, **kwargs)
      - asyncio.run_coroutine_threadsafe(coro, loop)
  - **Streams**
    - coroutine asyncio.open_connection(..)
    - coroutine asyncio.start_server(...)
    - **Unix Sockets**
      - coroutine asyncio.open_unix_connection(...)
      - coroutine asyncio.start_unix_server(...)
    - asyncio.StreamReader
      - coroutine read(n=- 1)
      - coroutine readline()
      - coroutine readexactly(n)
      - coroutine readuntil(separator=b'\n')
      - at_eof()
    - asyncio.StreamWriter
      - write(data)
      - writelines(data)
      - close()
      - can_write_eof()
      - write_eof()
      - transport
      - get_extra_info(name, default=None)
      - coroutine drain()
      - is_closing()
      - coroutine wait_closed()
  - **Subprocesses**
    - coroutine asyncio.create_subprocess_exec(...)
    - coroutine asyncio.create_subprocess_shell(...)
    - asyncio.subprocess.Process
      - coroutine wait()
      - coroutine communicate(input=None)
      - send_signal(signal)
      - terminate()
      - kill()
      - stdin
      - stdout
      - stderr
      - pid
      - returncode
  - **Exceptions**
    - asyncio.TimeoutError
    - asyncio.CancelledError
    - asyncio.InvalidStateError
    - asyncio.SendfileNotAvailableError
    - asyncio.IncompleteReadError
    - asyncio.LimitOverrunError
  - **Expressions**
    - await <awaitable>
    - async def <name>
    - async with <name>
    - async for <name>
  - asyncio.Task
    - cancel(msg=None)
    - cancelled()
    - done()
    - result()
    - exception()
    - add_done_callback(callback, *, context=None)
    - remove_done_callback(callback)
    - get_stack(*, limit=None)
    - print_stack(*, limit=None, file=None)
    - get_coro()
    - get_name()
    - set_name(value)
  - **Concurrency Primitives**
    - asyncio.Lock
      - coroutine acquire()
      - release()
      - locked()
    - asyncio.Event
      - coroutine wait()
      - set()
      - clear()
      - is_set()
    - asyncio.Condition
      - coroutine acquire()
      - notify(n=1)
      - locked()
      - notify_all()
      - release()
      - coroutine wait()
      - coroutine wait_for(predicate)
    - asyncio.Semaphore
      - coroutine acquire()
      - locked()
      - release()
    - asyncio.BoundedSemaphore
      - coroutine acquire()
      - locked()
      - release()
  - **Queues**
    - asyncio.Queue
      - maxsize
      - empty()
      - full()
      - coroutine get()
      - get_nowait()
      - coroutine join()
      - coroutine put(item)
      - put_nowait(item)
      - qsize()
      - task_done()
    - asyncio.PriorityQueue
      - maxsize
      - empty()
      - full()
      - coroutine get()
      - get_nowait()
      - coroutine join()
      - coroutine put(item)
      - put_nowait(item)
      - qsize()
      - task_done()
    - asyncio.LifoQueue
      - maxsize
      - empty()
      - full()
      - coroutine get()
      - get_nowait()
      - coroutine join()
      - coroutine put(item)
      - put_nowait(item)
      - qsize()
      - task_done()
    - Exceptions
      - asyncio.QueueEmpty
      - asyncio.QueueFull