class CustomPiece:

Data to store:
- piece type → (Stores piece & colour)
- blender_obj of piece type
  - Set colour & location in init
- reference to Array
- Current Array index (Piece location on board)

Array = [r, n, b, k, q, n, b, r,
p, p, p, p, p, p, p, p,
...
] blanks are None

ID with len=64
0 1 2 3 ...

Methods

- def move(Self, new_loc): (Self: Custom piece; new_loc: int)
  - update blender_obj location ← don't set keyframes here.
  - Set Array[new_loc] = Self
  - Set Array[current_loc] = None
  - Set current_loc = new_loc

  } moves piece. This order prevents garbage collection getting greedy

- def die(Self):
  - Animate capture, ie. move &/or explosion
  - Set Array[current_loc] = None ← deleting } maybe garbage collection??
  - return Self

Initialisation

for Square in range(64):
  if board.piece_at(square) is not None:
    look up blender source some how
    CustomPiece(board.piece_at(square), blender_obj, loc)

Steps:
move
Check capture
Check Promotion

* move
  Check castle
  True:
    queen side? king side?
      check colour? Square is then known.
      move both pieces. Animation?
  False:
    - Call move on Custom piece at move.from_square

* Capture
  Check en passant
    True:
      - Call die() on Custom piece at move.ep_square
    False:
      - Call die() on Custom piece at move.to_square
  Append to some Captured list
  Animate moving

TODO no plan

* Promotion
  remove old model unlink?
  Array[move.to_square] = CustomPiece(move.promotion, blender_obj)