Allocation of Objects



Robert Smallshire
COFOUNDER - SIXTY NORTH
@robsmallshire



Austin Bingham
COFOUNDER - SIXTY NORTH
@austin_bingham

Naming Special Functions

__feature___



dunder

Our way of pronouncing special names

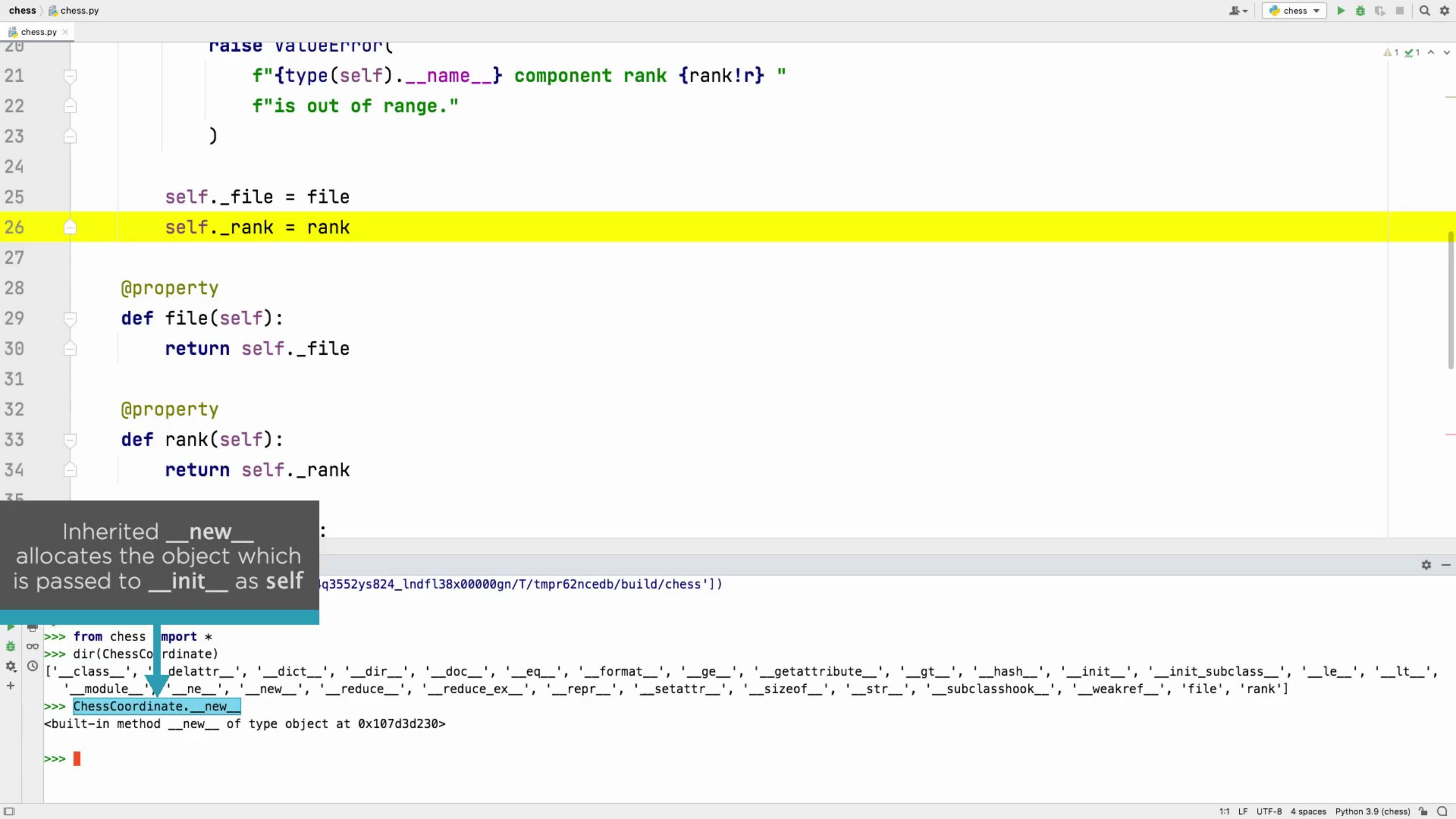
A portmanteau of 'double underscore'

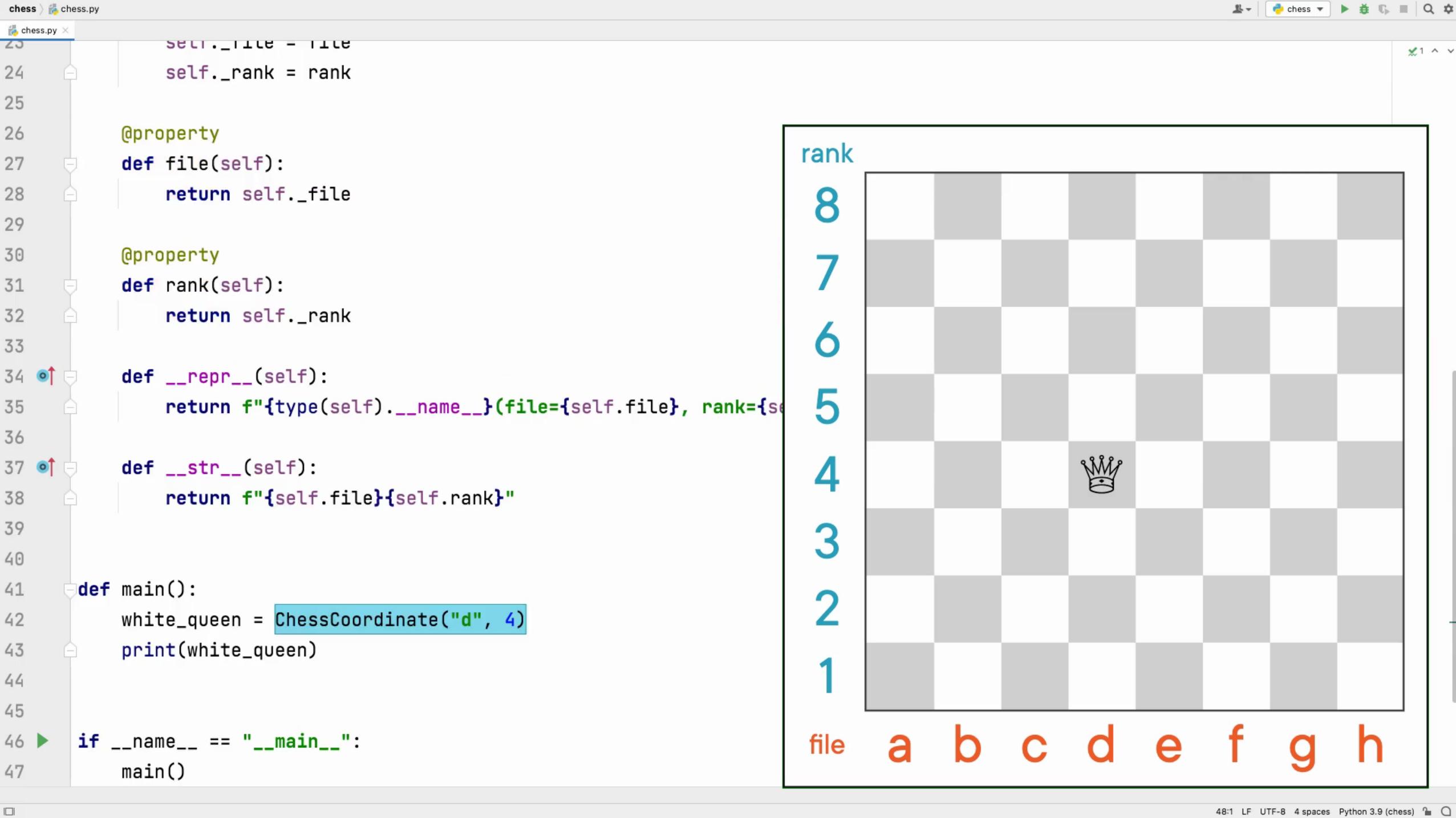
Instead of __init__ we'll say "dunder init"

Object Allocation



What does happen when you create an object?







More on object.__setatr__

Core Python: Custom Attributes and Descriptors

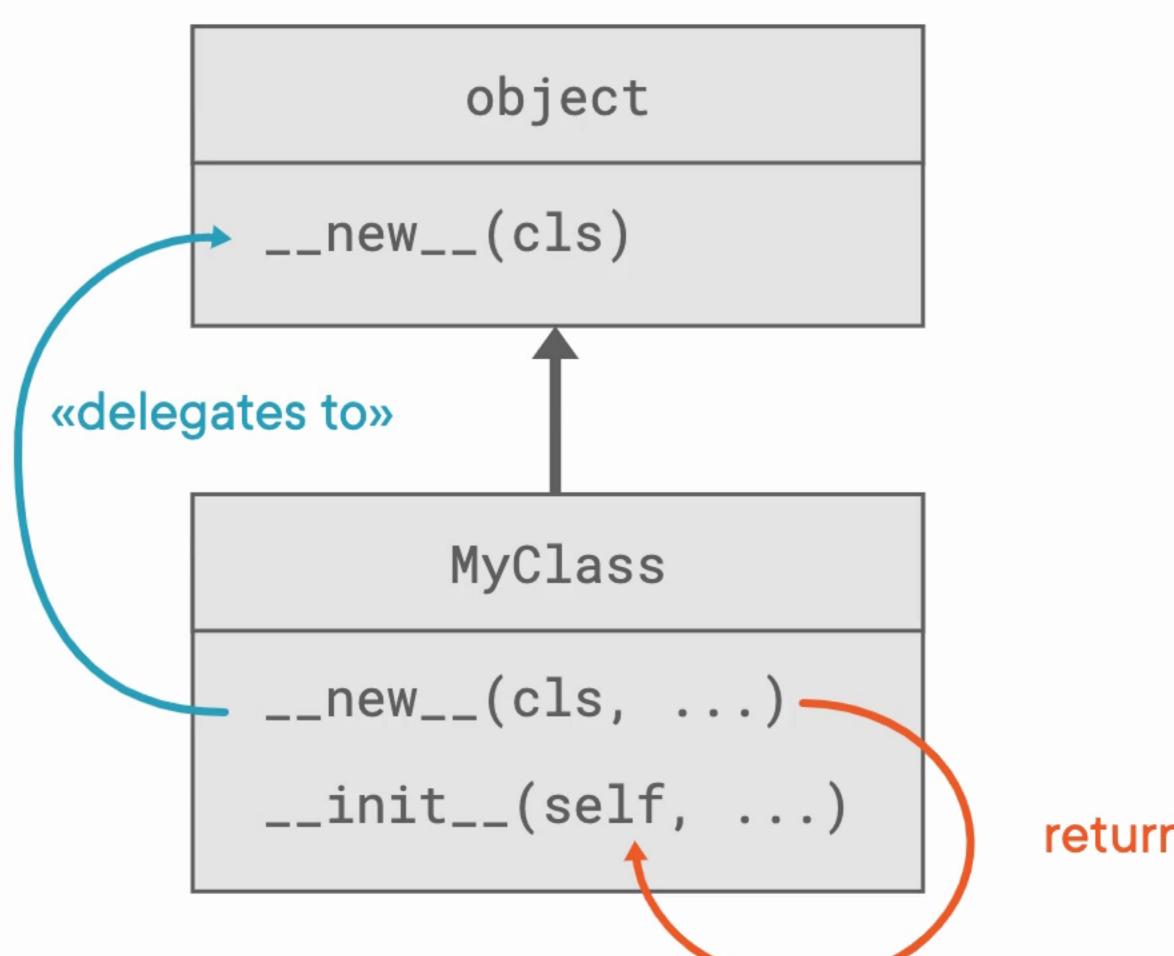
Robert Smallshire & Austin Bingham, Sixty North

Allocation with __new__



Customizing Allocation

Customizing Allocation



return value passed as self

Interning

Re-using objects of equal value on-demand instead of creating new objects

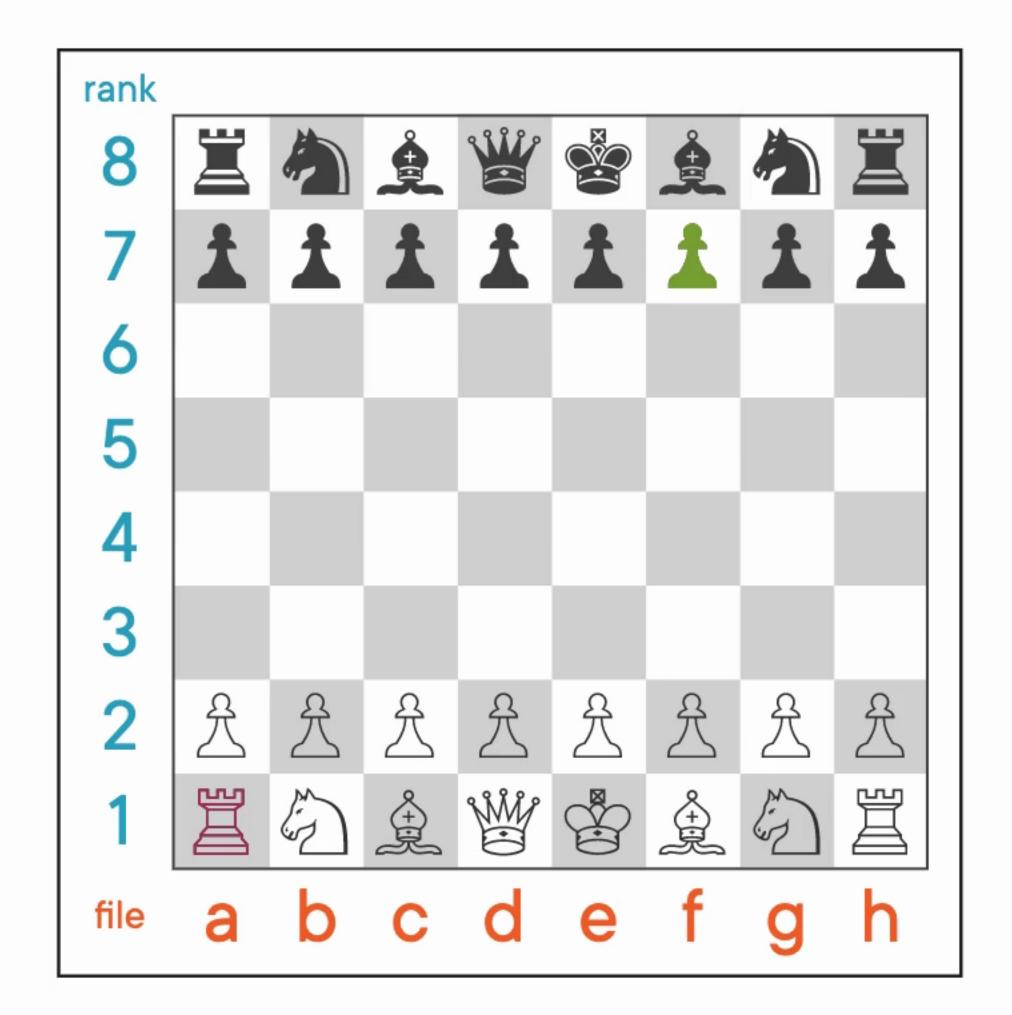
wikipedia.org/wiki/Interning

Chess Piece Encoding



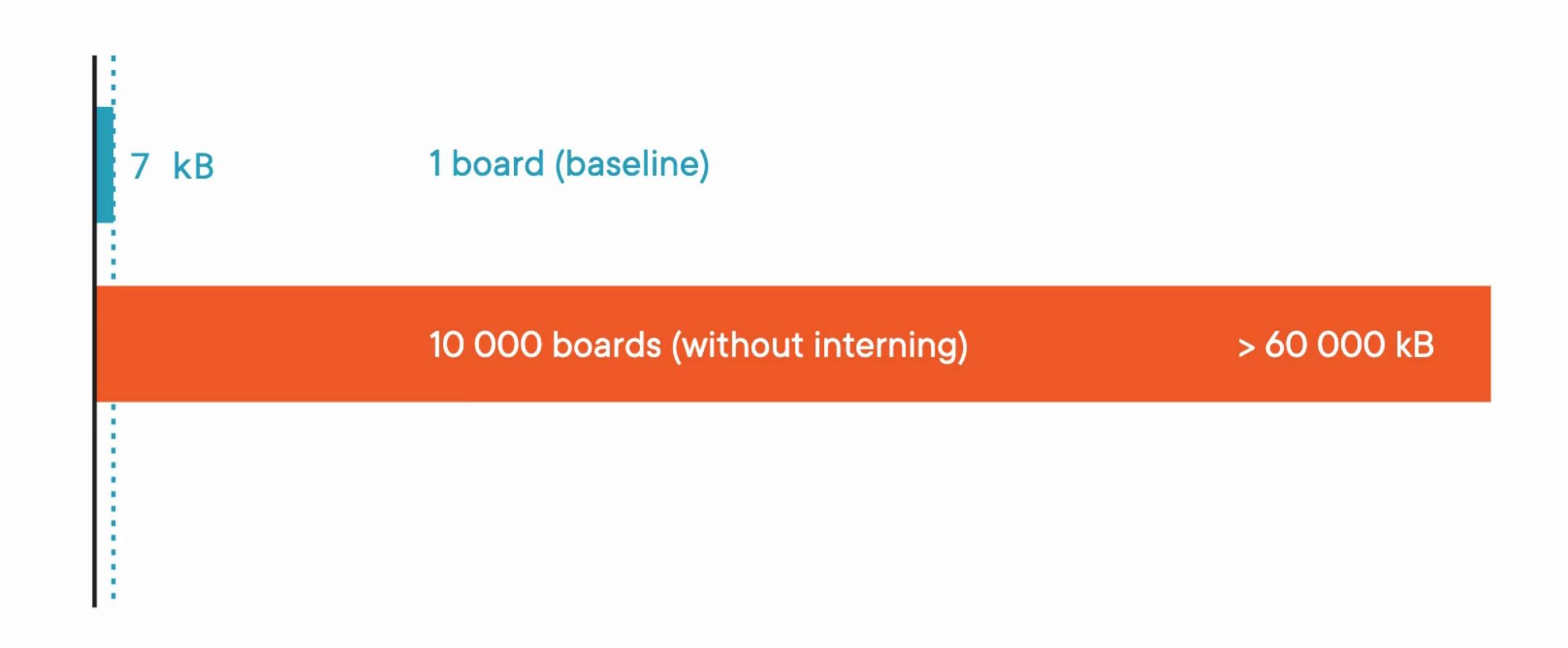
```
{
    '幽園': ChessCoordinate('a', 1),
    '幽色': ChessCoordinate('b', 1),
    '幽逸': ChessCoordinate('c', 1),
    '幽幽': ChessCoordinate('d', 1),
    '��': ChessCoordinate('e', 1),
    ...
    '��': ChessCoordinate('f', 7),
    '��': ChessCoordinate('g', 7),
    '��': ChessCoordinate('h', 7),
}
```



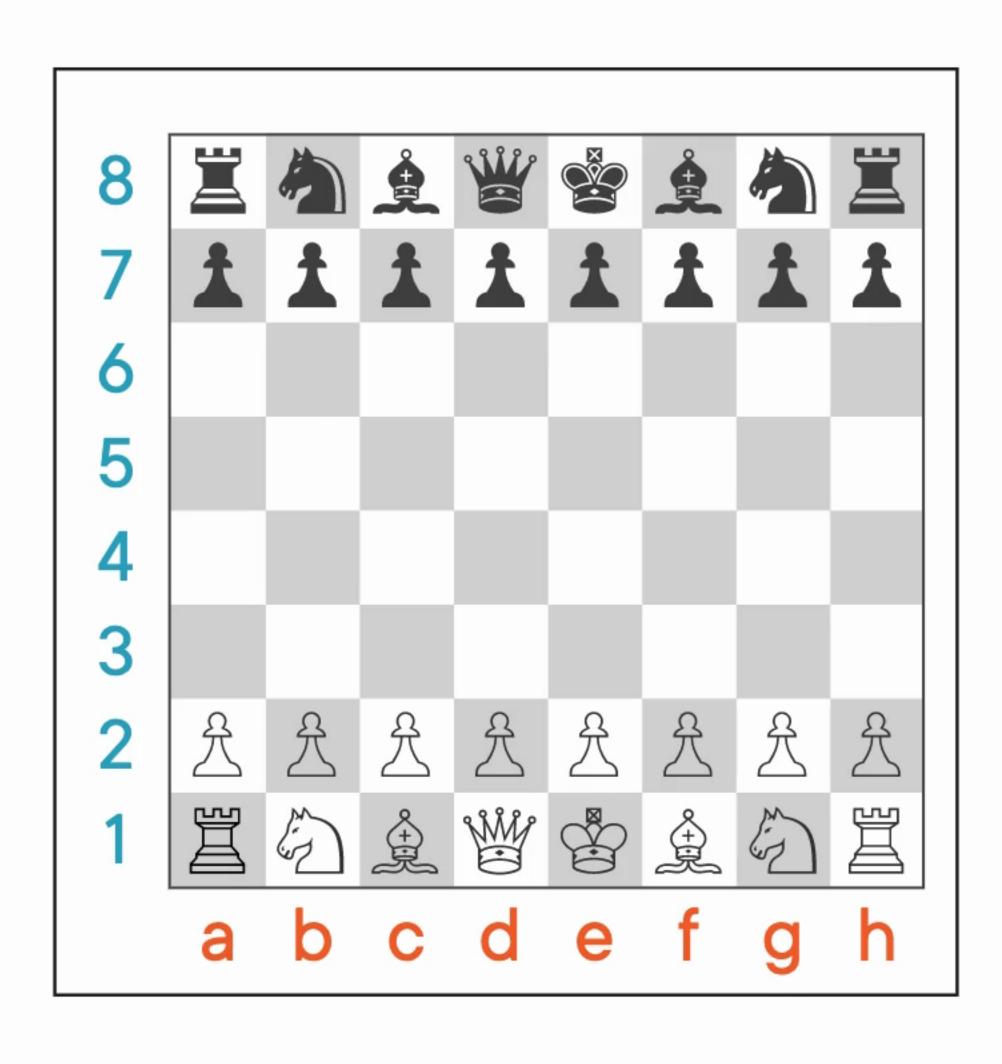




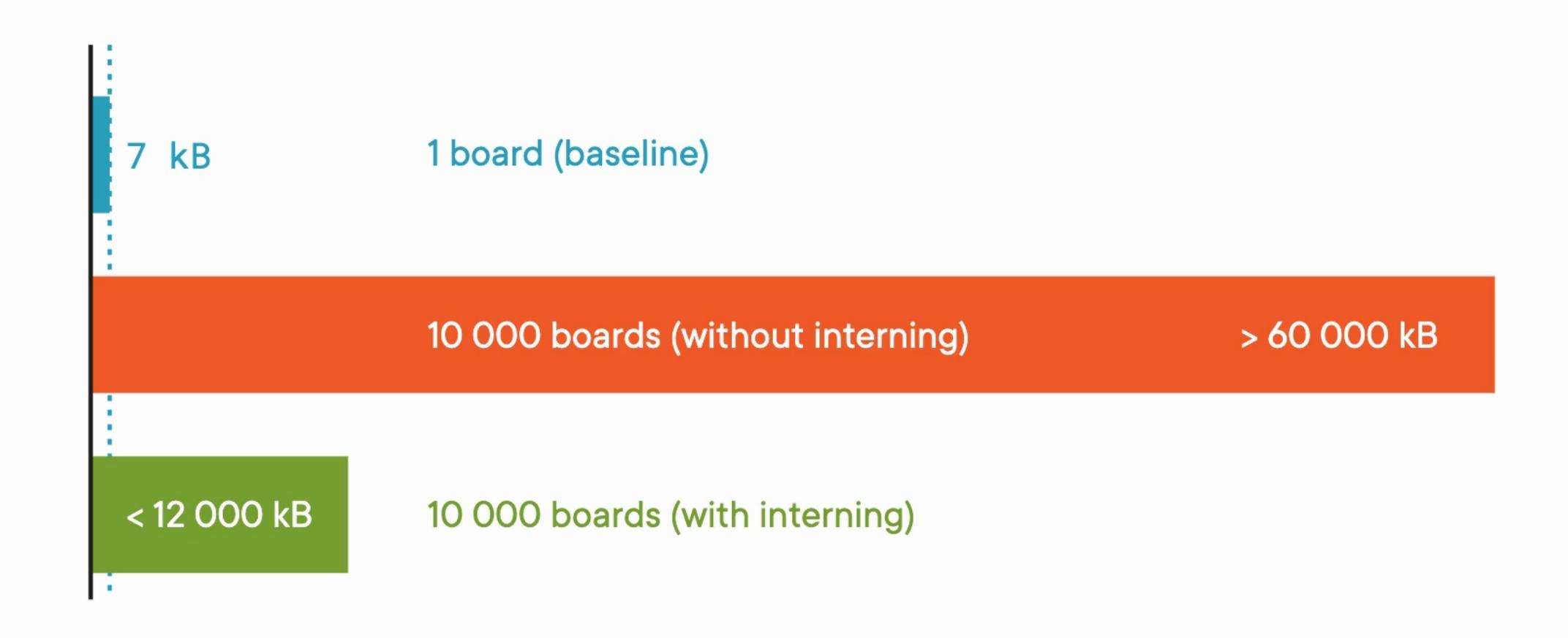
Chess Board Memory Usage



64 Positions, 32 Pieces



Chess Board Memory Usage



Interning

Only use for **immutable** value types

Summary



The special method __new__ allocates instances

The new instance is passed to __init__ as self

The ultimate allocator is object.__new__(cls)

Override __new__ to customize allocation

The __new__(cls) method is implicitly static

Interning reuses existing objects of equal value