CPython 原始碼解析

果凍

簡介

- 中興大學學士
- 任職於曼克斯
- 接觸 python 時間有七年
- 喜歡學習新的程式語言
- C、C++、java、golang。
- Linkin: http://www.linkedin.com/pub/kao-kuo-tung/67/9a2/6b3
- About me: http://about.me/ya790206

object.h

Python 物件根本:

1. PyObject
   a. 物件大小固定
   b. 如 int
   c. PyObject_HEAD

2. PyVarObject
   a. 多了 ob_size 欄位
   b. 如 string, list
   c. PyObject_VAR_HEAD

ref: http://docs.python.org/2/c-api/structures.html

```
#define PyObject_HEAD                   \
    _PyObject_HEAD_EXTRA                \
    Py_ssize_t ob_refcnt;               \
    struct _typeobject *ob_type;


#define PyObject_VAR_HEAD               \
    PyObject_HEAD                       \
    Py_ssize_t ob_size; /* Number of items in variable part */
```

```
typedef struct _object {
    PyObject_HEAD
} PyObject;

typedef struct {
    PyObject_VAR_HEAD
} PyVarObject;
```

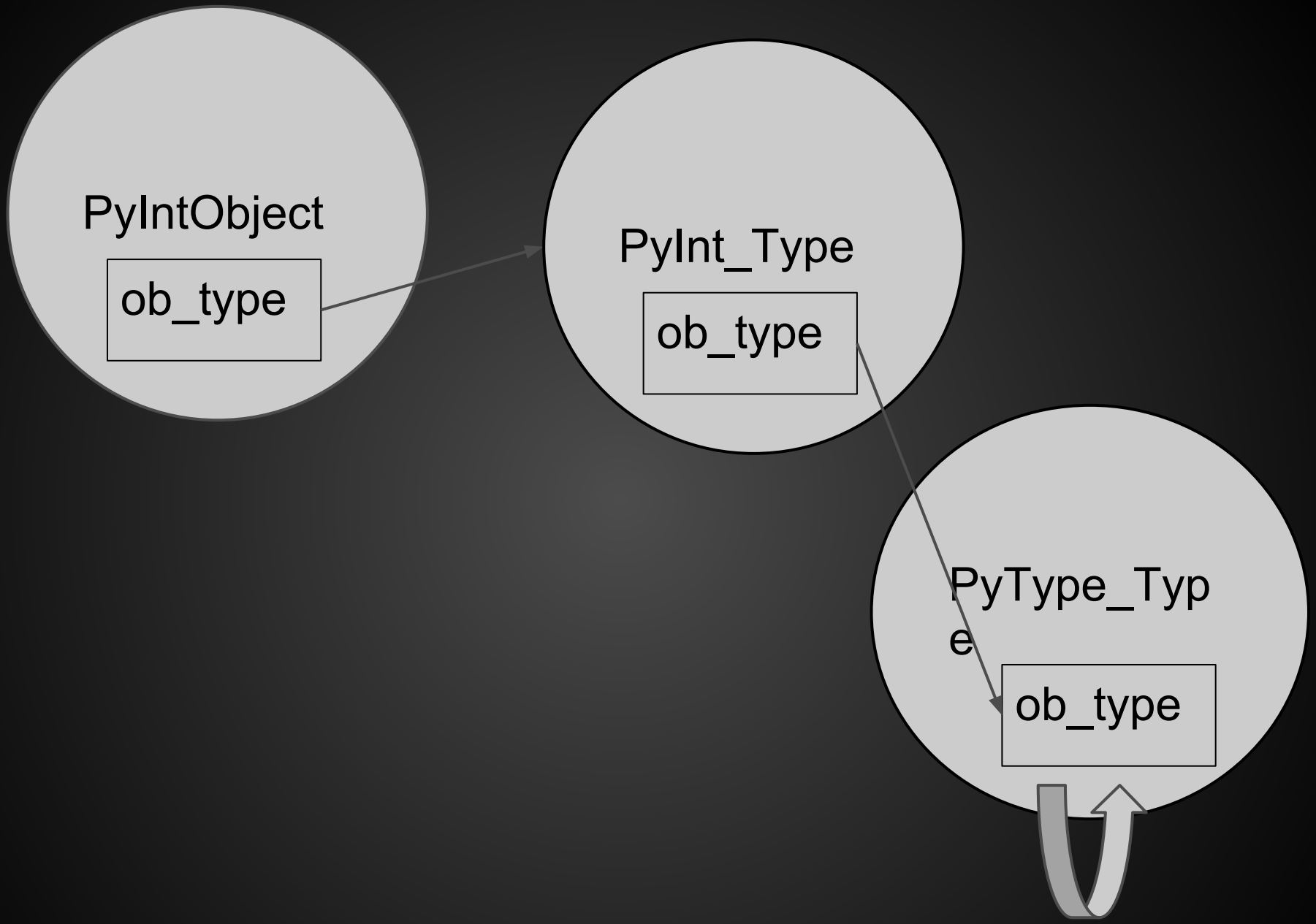
ob_refcnt:

1. Reference Counting

```
#define Py_INCREF(op) (                         \
    _Py_INC_REFTOTAL  _Py_REF_DEBUG_COMMA
\
    ((PyObject*)(op))->ob_refcnt++)
```

ob_type:

a. 該物件的 type
b. PyType_Type 物件的此欄位指向自己
c. 其他屬於 PyTypeObject 的物件此欄位指向 PyType_Type 物件
d. 其他物件則指向他所屬的 PyTypeObject 物件


PyIntObject
ob_type
PyInt_Type
ob_type
PyType_Type
ob_type

PyObject

1. PyClassObject
2. PyInstanceObject
3. PyMethodObject
4. PyCodeObject
5. Py_complex
6. PyDictObject
7. PyFileObject
8. PyFunctionObject
9. PyIntObject
10. PySetObject

等等

PyVarObject

1. PyByteArrayObject
2. PyFrameObject
3. PyListObject
4. PyStringObject
5. PyTupleObject

PyTypeObject

1. 存放該物件可以被 python 執行的方法

2. 如 PyInt_Type 存放 Int 型別所支援的方法，他支援 tp_str，所以我們可以使用 str(5)。當我們呼叫 str(5)，他會呼叫相對應的 c function, int_to_decimal_string。
3. 因為 tp_call 的值是 0，因此 int 型別不能被呼叫。tp_call 對應到 python 的 __call__ 方法。

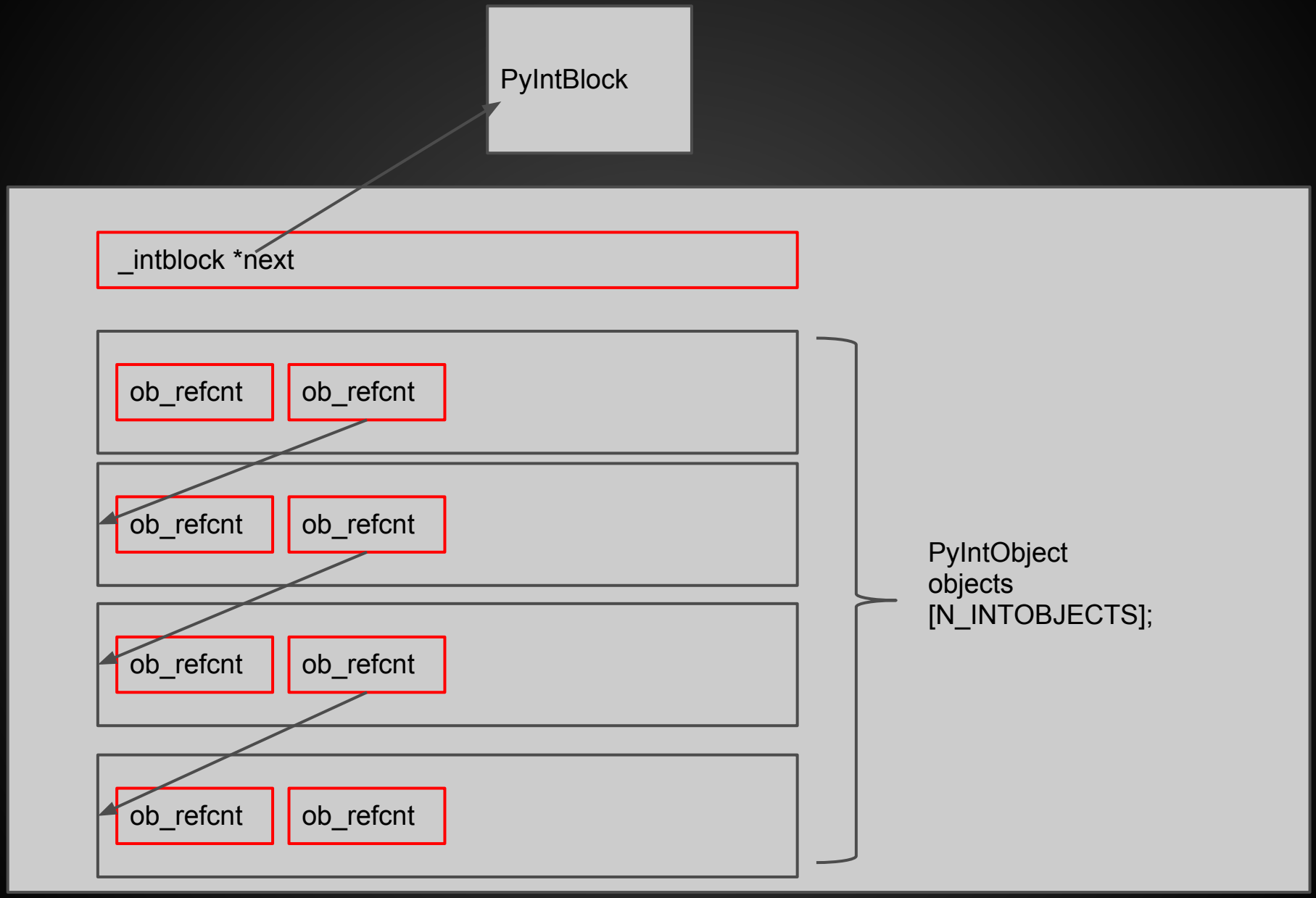
PyTypeObject

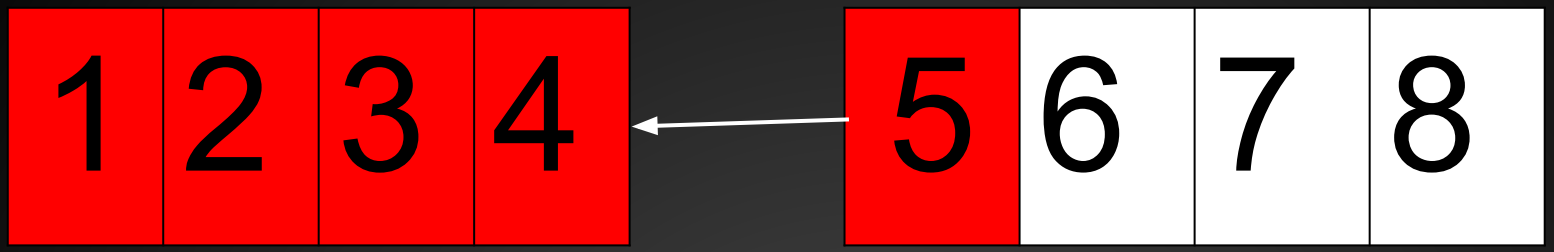
```
#define Py_TYPE(ob)           (((PyObject*)(ob))->ob_type)

PyObject *v;
Py_TYPE(v)->tp_free((PyObject *)v);
```

PyIntObject

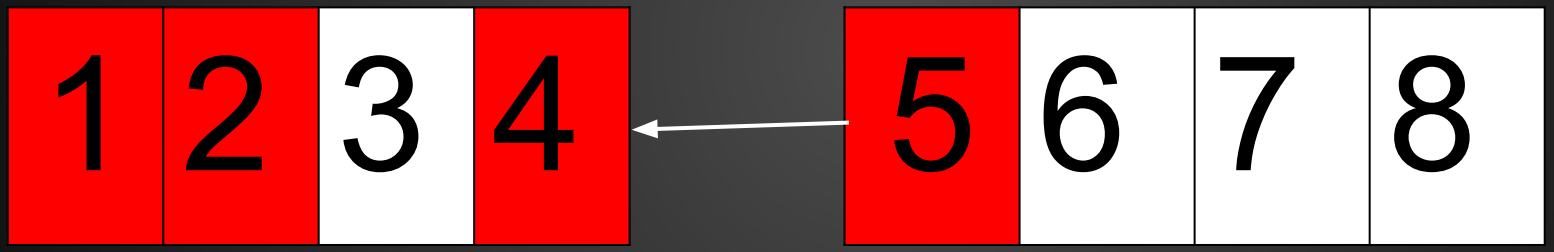
1. PyInt_FromLong(long ival) 建立整數的函數
2. 預設 CPython 實作, -5 ～ 256 的整數物件都是 singletons。
3. 使用 free_list 來減少沒必的 memory allocate/deallocate。
4. 每次向系統要求可容納 N_INTOBJECTS 個整數的空間。
5. 做連續 n 次加法時, 會產生 n - 1 個暫時物件。因 int_add 的回傳值是 PyObject。


PyIntBlock
_intblock *next
ob_refcnt
ob_refcnt
ob_refcnt
ob_refcnt
ob_refcnt
ob_refcnt
ob_refcnt
ob_refcnt
PyIntObject
objects
[N_INTOBJECTS];



free list: 6 -> 7 -> 8

如果在第三個位置的物件被刪除後



free list: 3 -> 6 -> 7 -> 8