Projekt iMalloc
Koddokumentation på gränssnittsnivå

Niclas Edenvin
Åke Lagercrantz
Andreas Lelli
Daniel Lindgren
Elias Lundeqvist
Jakob Sennerby

2012-11-09

1 iMalloc

Synopsis

```
#include imalloc.h
struct style *iMalloc(chunk_size memsiz, unsigned int flags);
```

Description

iMalloc returns a pointer to a struct with memsiz reserved memory. iMalloc behaves differently depending on which flags has been chosen, the flags chose which functions to call.

memsiz is the user defined memory size and the flags changes the way the memory is behaving and which functions to use.

memsiz

The number of bytes you want to reserve which can be entered in several forms.

`1Mb` - Reserves 1 megabyte of memory
`1Kb` - Reserves 1 kilobyte of memory
`10` - Reserves 10 bytes of memory
`sizeof(int)*10` - Reserves enough memory to store 10 integers

flags

Flags are entered separated by a plus sign. Eg. ASCENDING_SIZE+GCD The possible flags to choose from is listed below:

First Choose how the freelist should be sorted

`ASCENDING_SIZE` - Sort the list with small objects first, large objects in the end
`DESCENDING_SIZE` - Large objects first, small objects in the end
`ADDRESS` - Sort the list depending on their adress, low adresses first, higher towards the end

Second Choose which kind of memory manager to use (Note: only REFCOUNT and GCD can be combined)

`MANUAL` - Memory allocation using alloc and free
`REFCOUNT` - Managed memory allocation using reference counter
`GCD` - Managed memory allocation using the mark and sweep algorithm for garbage collection

Any other combinations will produce unspecified results and we cannot guaranty functionality in those cases.

Usage examples: Memory with a size of 2Mb, a freelist sorted after descending size and with garbage collection; `iMalloc(2Mb, ASCENDING_SIZE+GCD)`

Memory with a size of `sizeOf(int)*10`, a freelist sorted after adress and refcount combined with GCD; `iMalloc(sizeOf(int)*10, ADRESS+GCD+REFCOUNT)`

2 Structs

Synopsis

```
typedef struct {
  TypedAllocator alloc;
  Global         collect;
} GC;
```

Description

alloc is a pointer to a function used to allocate memory within the adress space created by the iMalloc function. collect is a pointer to a function used to start a garbage-collecting process according to the mark- and sweep algorithm.

Synopsis

```
typedef struct {
  Local       retain;
  Manipulator release;
  Local       count;
} Refcount;
```

Description

This struct is used if a memory object is using refcount. retain is used to increment the refcount release is used to decrease the refcount count returns the current refcount value

Synopsis

```
typedef struct {
  RawAllocator alloc;
  Global       avail;
  Manipulator  free;
} manual, *Manual;
```

Description

This struct is used if using manual memory manager. alloc is a pointer to a function used to allocate memory within the adress space created by the iMalloc function. avail returns the total size of the free space in the address space. free frees an object in memory mem and returns the amount of memory freed.

Synopsis

```
typedef struct {
  RawAllocator alloc;
  Refcount     rc;
  GC           gc;
} managed, *Managed;
```

Description

This struct is used if using managed memory manager. alloc is a pointer to a function used to allocate memory within the adress space created by the iMalloc function. rc is set when using refcount and/or gc is set when using refcount

```
typedef union {
  manual  manual;
  managed managed;
} style;
```

Description

DESCRIPTION HÄR!!!!!

3 Typedefs

```
typedef struct style *Memory;
typedef void *(*RawAllocator)(Memory mem, chunk_size size);
typedef void *(*TypedAllocator)(Memory mem, char* typeDesc);
typedef unsigned int(*Manipulator)(Memory mem, void *ptr);
typedef unsigned int(*Global)(Memory mem);
typedef unsigned int(*Local)(void *ptr);
```