717310: Game Programming

Steffan Hooper

Thursday, 30 July 2015

Overview

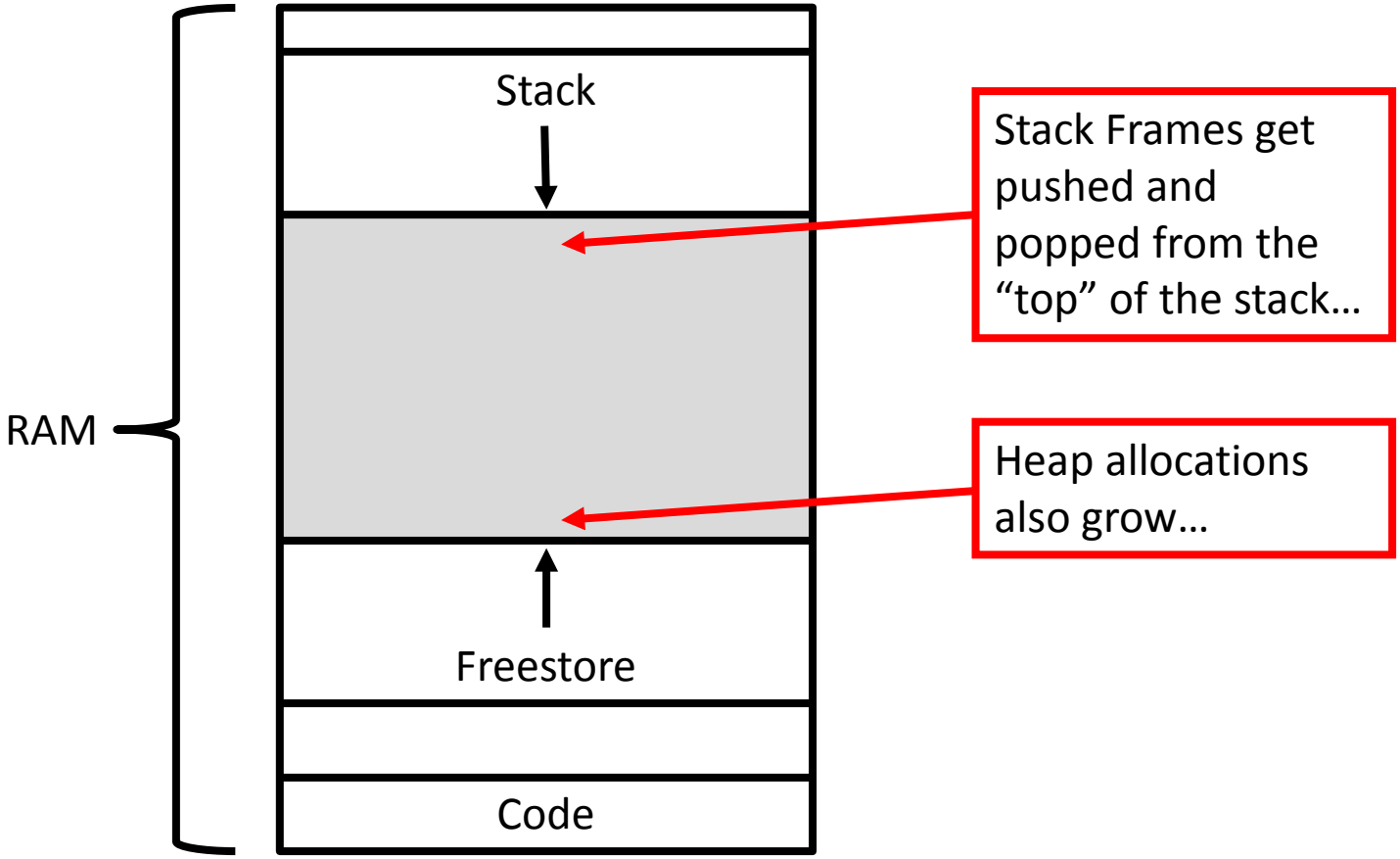
- More C++
  - Memory: The Stack vs The Freestore
  - Memory: Debugging
  - The Freestore
  - Using Freestore Arrays
- SiSo
- Exercises

Console Game Programming

- C++: Memory Management
  - Global Variables:
    - Declared outside of functions.
    - Accessible anywhere in the program.
  - Local Variables:
    - Declared in a function.
    - Accessible only in the function.
  - Parameters:
    - Passed into functions.
    - Stored in the stack frame.

Console Game Programming

- C++: Memory Management



Console Game Programming

- C++: Memory Management continued…
  - Stack Frames:
    - Allocated when the function is called.
    - Deallocated when the function returns.
    - Visual Studio Debug Builds:
      - Stack allocated but uninitialised local variables hold the value of **`0xCCCCCCCC`**
    - Beware of writing to stack variables, it is possible to trample the stack if you write beyond the bounds of a local array…

Console Game Programming

- C++: Memory Management continued…
  - The Freestore: Dynamic Memory.
    - Allocated using the **new** keyword.
    - Allocations persist until explicitly freed.
    - Must be deallocated using the **delete** keyword.
    - Visual Studio Debug Builds:
      - Allocated but unitialised Freestore variables hold the value of: **0xCDCDCDCD**. “Clean Memory”
      - Deallocated freestore allocations hold the value of: **0xFEEEFEEE**. “Freed Memory”
      - The boundaries of the allocaiton are marked with: **0xFDFDFDFD**. “Fence Guard – No Man’s land”

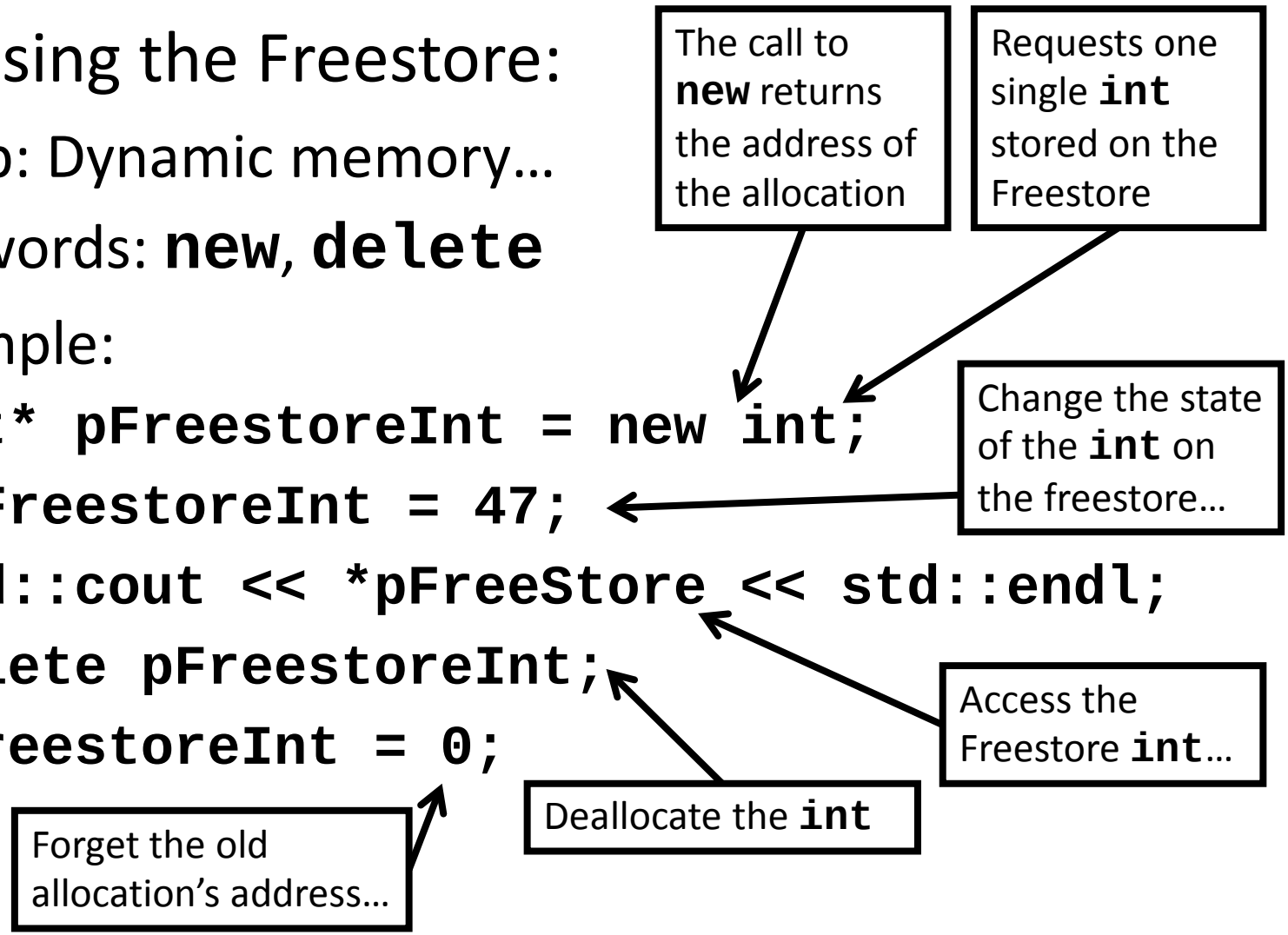
Console Game Programming

- C++: Memory Management continued…
  - C++ is not a managed language…
    - This means the programmer must explicitly manage the allocation and deallocation of memory.
  - Forgetting to free dynamic memory once it is no longer needed will cause a memory leak.
    - Memory leaks mean you may run out of RAM to store data in your program.
    - Beware… only keep data around as long as you need it!
    - Think carefully about data's lifecycle.

Console Game Programming

- C++: Using the Freestore:
  - Heap: Dynamic memory…
  - Keywords: **new**, **delete**
  - Example:

```
int* pFreestoreInt = new int;
*pFreestoreInt = 47;
std::cout << *pFreeStore << std::endl;
delete pFreestoreInt;
pFreestoreInt = 0;
```



Console Game Programming

- C++: The Freestore continued…
  - Dynamic memory allocation:
  - Keywords: **`new []`**, **`delete[]`**
  - Example:

```
int* pArray = new int[10];
pArray[0] = 110;
// ...
pArray[9] = 990;
delete[] pArray;
pArray = 0;
```

Requests ten **`int`** variables in a contiguous blcok on the Freestore

The call to **`new`** returns the address of the 0'th element

Access the elements like a normal array…

Deallocate the array when no longer needed…

Console Game Programming

- C++: The Freestore continued…
  - Beware!
  - Every **`new`** must be paired with a **`delete`**…
    - At some point in your program, you must deallocate the memory…
      - Otherwise there will be a memory leak.
  - Every **`new []`** must be paired with a **`delete[]`**
    - As above…
  - And: Do not mix **`new []`** with **`delete`**
  - And: Do not mix **`new`** with **`delete[]`**

Console Game Programming

- C++: Using Arrays…
  - Usually you do the following to access an array element:
    - **`arrayName[0] = 10;`**
    - This assigned the value of 10 to element index 0.
  - What this actually does is:
    - **`*(arrayName + 0) = 10;`**
    - Pointer arithmetic!
      - **`arrayName`** stores the pointer to the first element… add the index, dereference that location, and then write to that location in memory!

Console Game Programming

- C++: Using Arrays strangely…
  - This means you can actually do weird syntax such as:
    - **`5[arrayName] = 25;`**
    - This assigned the value of 25 to element index 5 in the **`arrayName`** array.
  - What this actually does is:
    - **`*(5 + arrayName) = 25;`**
    - Pointer arithmetic!
    - Strange… but it works…

Be a good programmer and put the index inside the **`[ ]`** brackets… don't confuse your colleagues!

Console Game Programming

- The C way of allocating Dynamic Memory…
  - **`malloc`**
    - Allocates on the Heap, returns a **`void*`**
  - **`free`**
    - Deallocates a Heap allocation.
  - Do not mix **`malloc`** and **`delete`**.
  - Do not mix **`new`** and **`free`**.
  - Use the C++ Dynamic Memory calls…
    - These will invoke constructors, **`malloc`** will not!

Console Game Programming

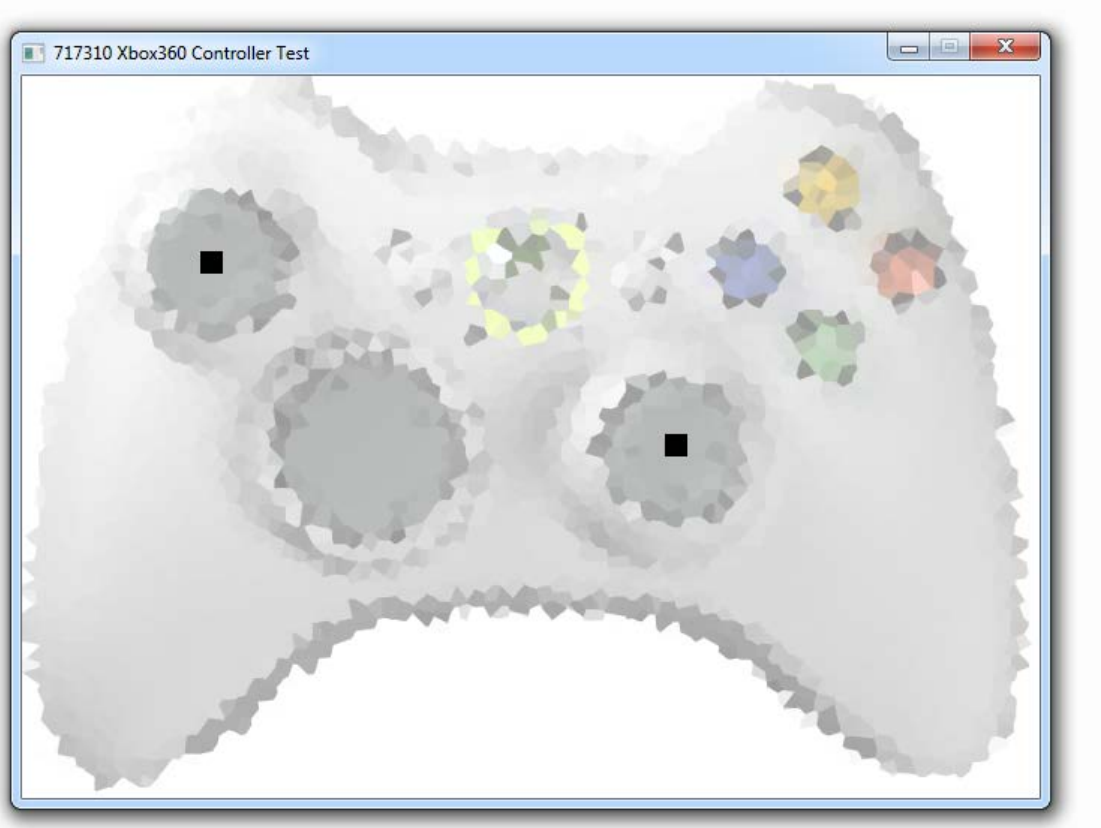
- C++: Casting
  - C-Style: **`(int) 3.14f`**
  - **`reinterpret_cast<`*`type`*`>(`*`value`*`)`**
    - Converts type to incompatible type…
  - **`static_cast<`*`type`*`>(`*`value`*`)`**
    - Converts one type to another compatible type…
  - **`dynamic_cast<`*`type`*`>(`*`value`*`)`**
    - Converts virtual base class to derived class…
  - **`const_cast<`*`type`*`>(`*`value`*`)`**
    - Converts to compatible type with different “constness”

Log in to SiSo

- Go to: **`https://siso.aut.ac.nz/`**
- Choose:
  - Design and Creative Technologies (DCT)
- Login:
  - Using your AUT Username and Password.
- Review:
  - Your account and paper details…
  - And the Terms, Conditions and Loan Out Facility Rules. – Then click “Submit”

Xbox 360 Wired Controller

- Driver: **`Xbox360_64Eng.exe`**
- Xbox 360 Controller Test Program:



Exercises

- Week 2:
  - Day 004.1 – C++: Using References
  - Day 004.2 – C++: Using Structures
  - Day 004.3 – C++: Using Pointers and the Freestore

Exercises

- Recommended Readings:
  - Brownlow, M. (2004). *Game Programming Golden Rules*. Hingman, MA: Charles River Media, Inc.
  - Meyers, S. (2005). *Effective C++: 55 Specific Ways to Improve Your Programs and Designs* (3rd ed.). Reading, MA: Addison-Wesley Professional.
  - Sutter, H., & Alexandrescu, A. (2004). *C++ Coding Standards: 101 Rules, Guidelines, and Best Practices*. Upper Saddle River, NJ: Addison-Wesley Professional.

Summary

- SiSo
- More C++
  - Memory: The Stack vs The Freestore
  - Memory: Debugging
  - The Freestore
  - Using Freestore Arrays
- Exercises