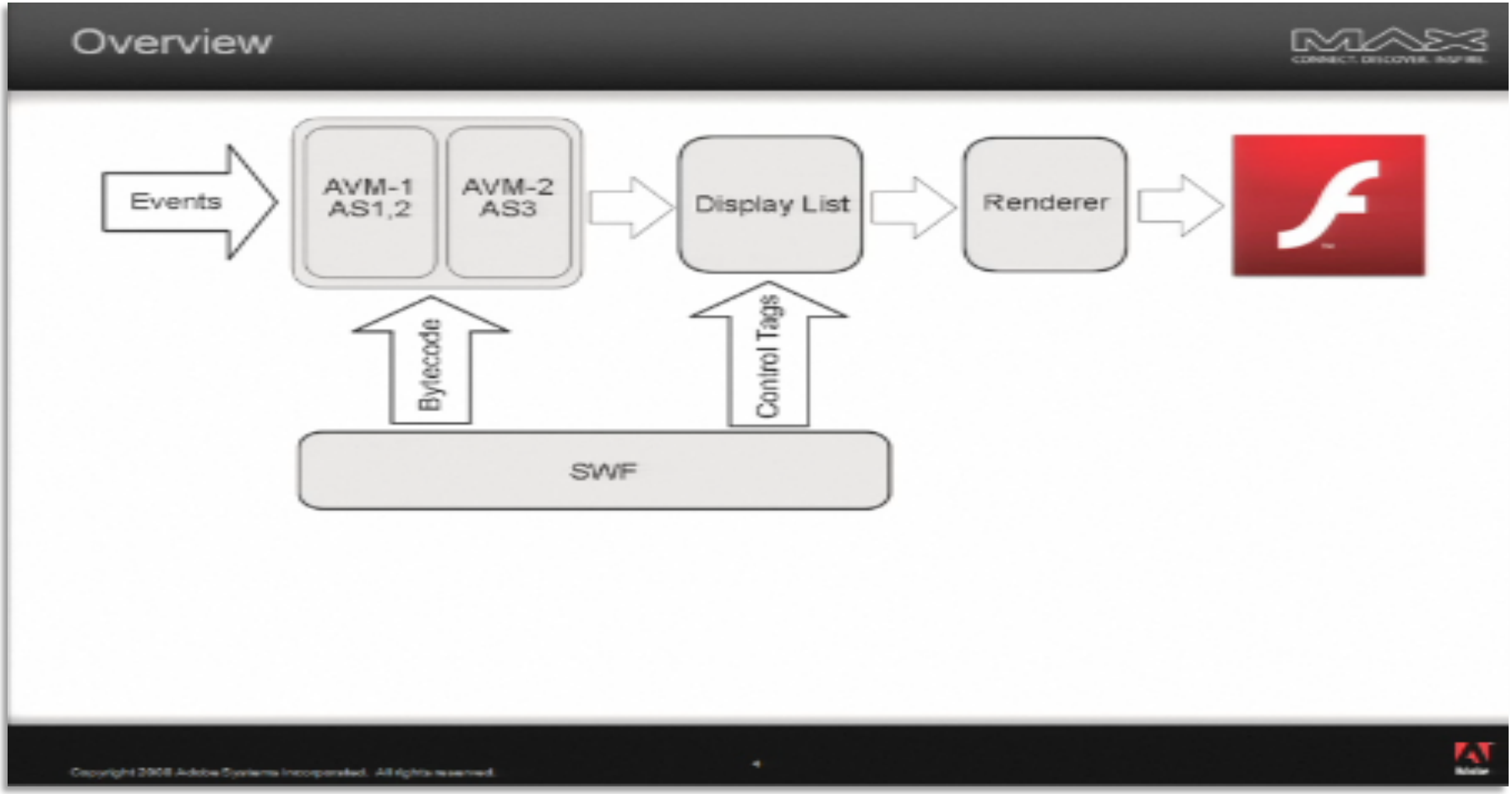
Flash Player Advanced

Presented By
Ashish Desai

The Birth Of Flash Player

- Began with FutureWave Software (a drawing application) co-created by Jonathan Gay and Charlie Jackson.
- Animations added to FutureWave and called it FutureSplash.
- They Approached Adobe in 1995 to sell FutureSplash but Adobe turned them down.
- Macromedia acquired FutureSplash in 1996, contracting the name to Flash (F~~utureSp~~lash).
- The Flash Player was born in 1996.
- Flash became Adobe Flash as a result of Adobe's acquisition of Macromedia.

Flash Player Internals



Flash Player Internals (Cont.)

- Flash Player is Single Threaded.
- Events are captured by Flash Player itself.
- No code can modify the Display List and Rendering component at the same time.
- Stage is the drawing board area for Flash. Stage is parent of everything you see in Flash Player in browser or standalone.
- Flash treats everything as a movie which it plays synchronizing itself with a play head (more on this in a bit).
- What is the default frame rate for a flex application?
  - 24 Frames Per Second
  - Typically a browser can produce 60 fps even if rate is set to 200 fps.

The Virtual Machine – AVM2/AVM1

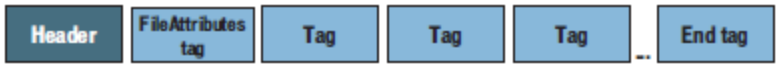
- Flash Player ships with two virtual machines (runtimes) – AVM1 and AVM2.
- AVM2 supports Action Script 3.0 specification.
- AVM1 is for the backward compatibility with Action Script 2.0 and Action Script 1.0 specifications.
- AVM interprets the ActionScript Byte Code (ABC) in the compiled SWF file into machine language.
- A Just-in Time Compiler (JIT) is included with AVM2 which converts similar looking byte code to machine language thus improving runtime performance.
- Constants, literals and numbers are JIT'ed always.

Quiz

- What does SWF stand for?
- The SWF (pronounced “swiff ”) stands for Shock Wave Flash & is a file format that delivers vector graphics, text, video, and sound over the Internet and is supported by Adobe® Flash® Player software. The SWF file format is designed to be an efficient delivery format, not a format for exchanging graphics between graphics editors.
- Why are flex applications fast (SWF Performance)?
  - The files are compressed to be small and support incremental rendering through streaming.
  - The SWF file format is a binary format and is not human readable like HTML.
  - The SWF file format uses techniques such as bit-packing and structures with optional fields to minimize file size.
  - The graphics described by SWF files render quickly because only the calculations are transmitted rather then pixel by pixel rendering.
  - Fonts, images and other external resources can be embedded without substantially increasing files size and hence lesser download times.
  - SWF files are downloaded just once (incremental rendering) and run local in the Flash Player unlike HTML.

SWF File Internals

- The structure of a SWF File



SWF File Structure

- Header Tag defines version of SWF, compression, frame rate etc.
- Two kinds of tags – Definition Tags & Control Tags
- Definition Tags define contents (Display Objects) of the SWF File. Each definition tag is assigned a unique ID and stored in a Dictionary.
- A tag can refer to a tag before it but in no case refer to a tag after it.
- Control tags manipulate the Display List. Adding, modifying and removing items from the Display List.
- Control tags create and manipulate rendered instances of items in the dictionary, and control the flow of the file.

SWF File Internals (Cont.)

- Flash Player processes all of the tags in a SWF file until a ShowFrame tag is encountered.
- At this point, the Display List is copied to the screen and Flash Player is idle until it is time to process the next frame.
- What you see in Flash Player is the cumulative effect of the definition tags and control tags until the ShowFrame tag is encountered.
- ShowFrame tag is inserted by the compiler in the compiled SWF.

Rendering

- Modes of rendering:
  - Immediate Mode Rendering
  - Retained Mode Rendering
- What mode does Flash Player use?
  - Retained Mode Rendering.
- What facilitates Retained Mode Rendering in Flash?
  - The Display List & Timeline bound rendering

The Display List

- The HEART of the Flash Player.
- Display List facilitates the retained mode rendering in Flash.
- Retained mode improves performance. Consider following code and that it happens in the same frame.

```
var button:Button() = new Button();
addChild(button)
removeChild(button);
addChild(button)
```

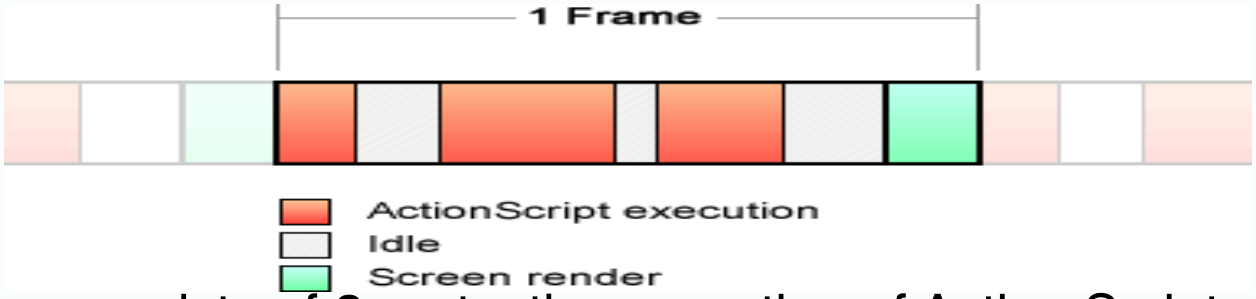
- Flash players display list will render the cumulative effect of the above code which is addition of button to its parent.
- Effectively avoided two instructions improving performance.

Display List - How it all works

- All Display Objects defined in the SWF file are stored in a Dictionary.
- Each object(item) is given a unique ID in the dictionary.
- Flash Player determines which items need to be displayed in the next frame.
- These items are selected from dictionary and placed on the Display List.
- 50% of each frame is dedicated to rendering, during which these items will be rendered.
- Changes to the dictionary and the Display List are controlled via various control tags that the compiler adds.

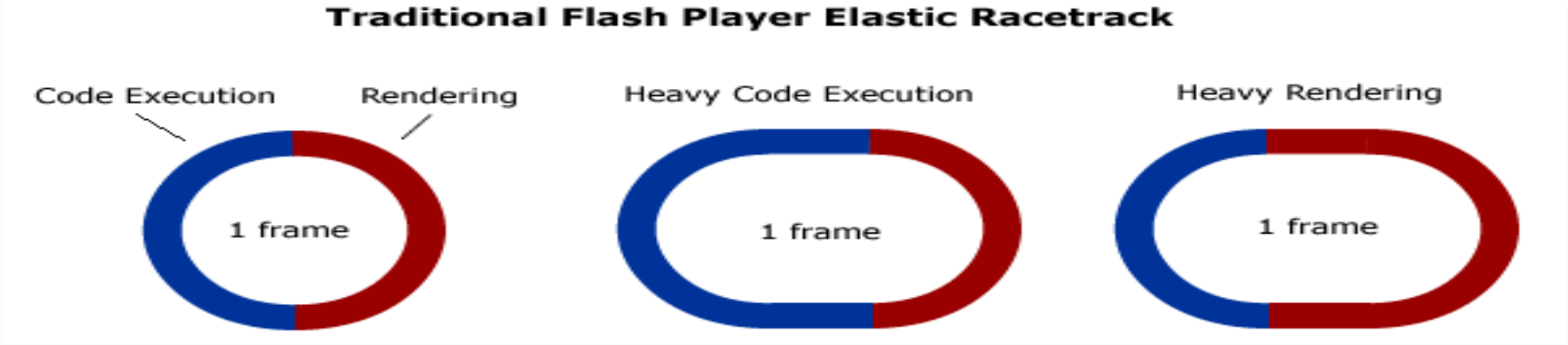
Flash Timeline – Elastic Racetrack

- Flash Player instance processes a constant loop of consecutive frames.



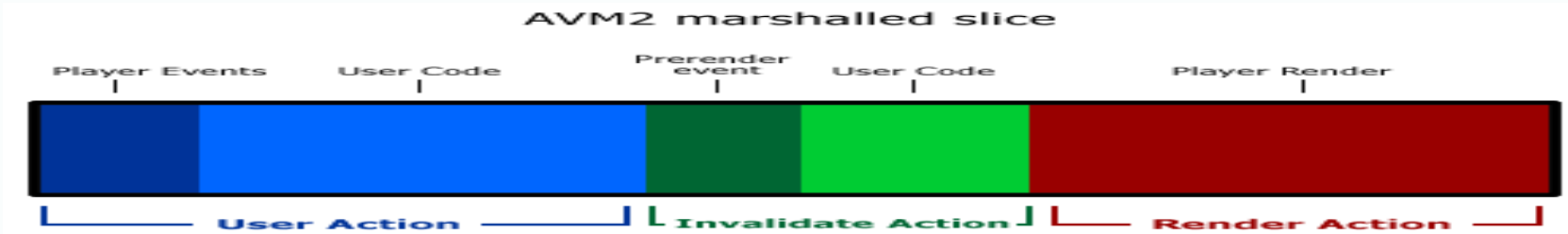
- Each frame consists of 2 parts: the execution of Action Script and a visual rendering of the screen by the Flash Player renderer.
- The execution of CPU-intensive Action Script increases time between each frame render.

The Elastic Racetrack (Cont.)

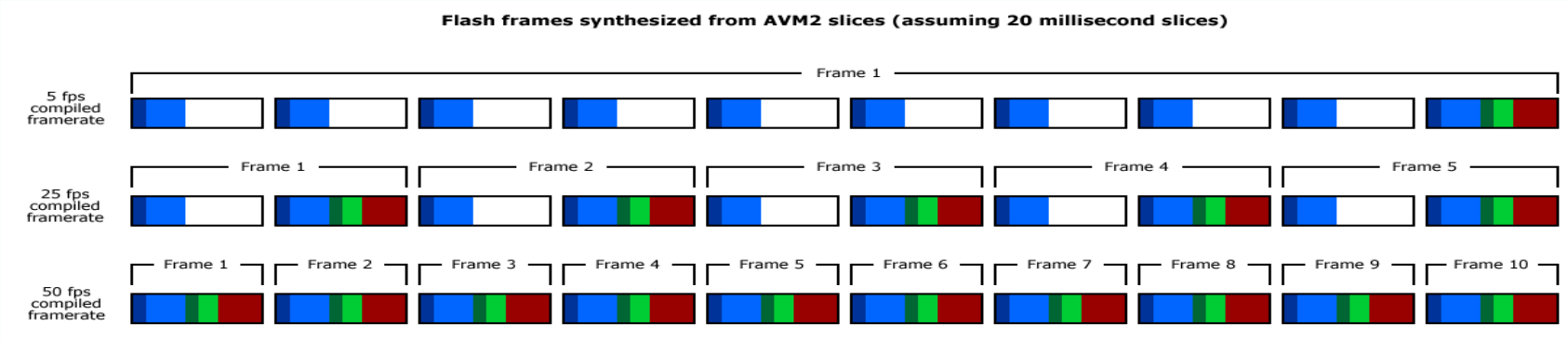


- If the combined operations of the frame loop (code execution and rendering) take too long, the runtime isn't able to maintain the frame rate.
- When the frame rate slows, the experience degrades. Animation becomes choppy. In worse cases, the application freezes and the window goes blank
- Debug player will given an exception "A script has taken more than X seconds to execute".

Time Slicing - Marshall



- Typically a 20ms slice within a frame. The Marshall synthesizes frame rates.



- Slicing at various frame rates. Player decides on when to render or execute script.

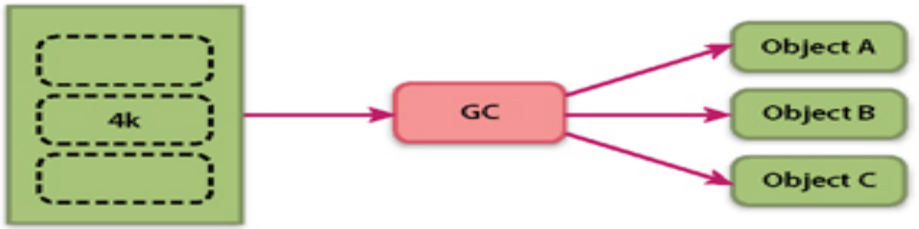
What does it mean? (Cont.)

- Avoid writing time consuming (CPU Intensive) action scripts
- User Timers carefully.
- Beware of infinite loops. Dispatch events with caution.
- Be careful while deeply nesting containers and Display Objects within containers.
- Use Binding with caution. Avoid Bindings when direct assignments are possible.
- Practical Example:
    - An infinite loop in Account to Trading Navigation froze Flash Player.
- As frame expands to accommodate and continuously repeats the frames its called “The Elastic Racetrack”

Memory Allocation

- Flash Player uses a page allocator (GCHeap) to allocate large blocks (megabytes) of memory from the OS.

- The GCHeap then breaks the large block into smaller 4K pages and gives the pages to the MMgc.
- The GC then uses those 4K pages to provide memory for objects up to 2K in size in the system.



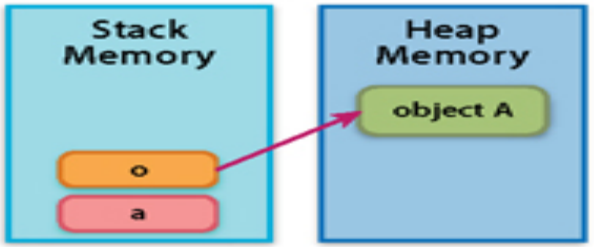
Memory Allocation (Cont.)

- For objects larger than 2K (bitmaps, videos, files, etc.), GCHeap provides groups of contiguous 4K blocks to a large memory allocator.
- When almost all of the 4K pages in a large chunk are allocated, Flash Player runs garbage collection to reclaim unused memory before the GCHeap attempts to allocate more memory from the OS.

Heap & Stack

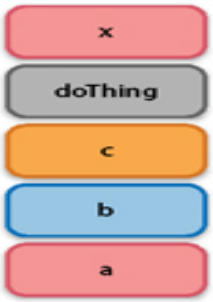
- The heap is the memory allocated for any object created or initialized at runtime. Objects on the heap exist until they are garbage collected.

```
function doThing() {
        var a:int;
        var o:Object;

        o = new Object();
}
```



- The stack is memory that stores all variables that are defined at compile time. Stack memory is used and reused in a sequential manner.

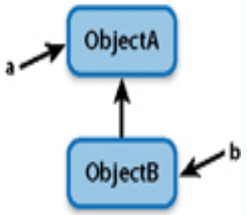
```
function doThing() {
        var a:int;
        var b:int;
        var c:Number;

        someMethod();
}
funtion someMethod() {
        var x:int;
}
```



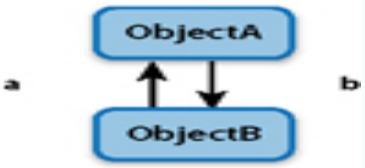
Runtime Garbage Collection

- Flash Player and AIR use a combination of deferred reference counting and conservative mark-and-sweep.
- Deferred Reference Counting
  - Each object on the heap keeps track of the number of things pointing to it.
  - If the object has a zero reference count (nothing is pointing to it), it is added to the Zero Count Table (ZCT) .

```
a = new ObjectA();
b = new ObjectB();
b.prop = a;

Reference counts:
   ObjectA: 2
   ObjectB: 1
```




```
a = new ObjectA();
b = new ObjectB();
b.prop = a;
a.prop = b;
a = b = null

Reference counts:
   ObjectA: 1
   ObjectB: 1
```

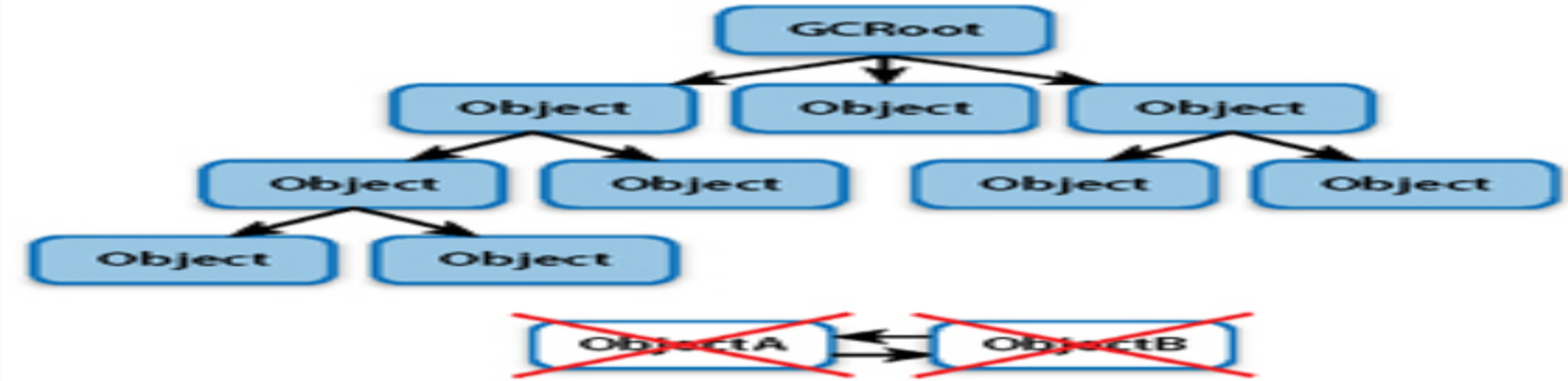


Runtime Garbage Collection (Cont.)

- When the ZCT is full, the stack is scanned to find any references from the stack to an object on the ZCT.
- Any object on the ZCT without a stack reference is deleted.
- Circular reference OR Islanded objects are not GC's using deferred Reference Counting.
- A more sophisticated MARK AND SWEEP algorithm takes care of it.
- [mark] and [sweep] can be seen in Flex Profiler.
- Flash Player pauses completely while it is Garbage Collecting.

Mark and Sweep

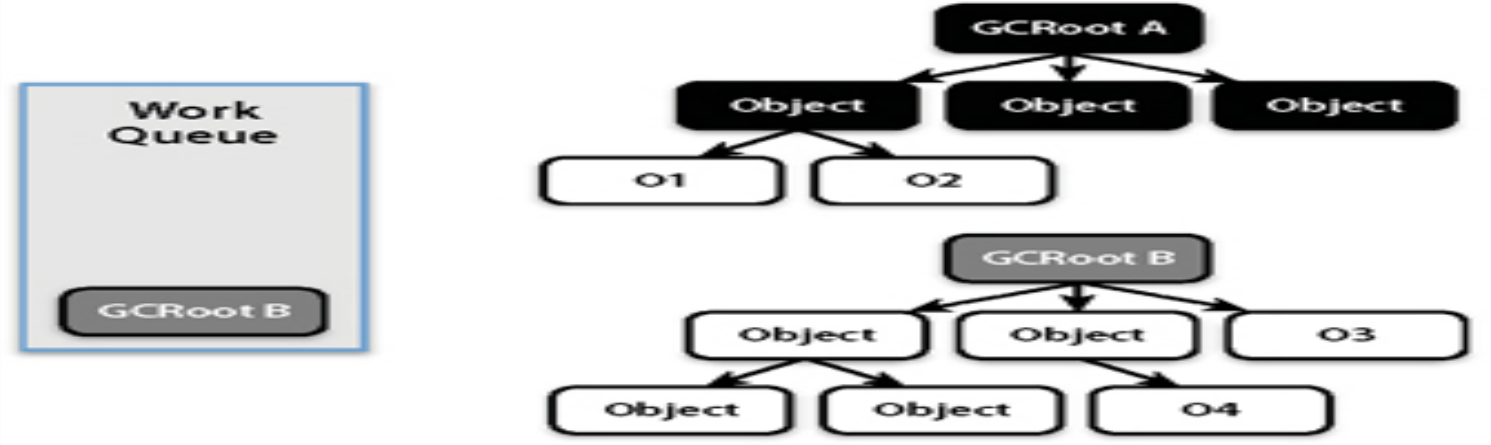
- An algorithm that traces an object from one of its GC roots.
- GCRoots are never garbage collected.



- All objects in flash have a mark bit. During Mark phase all objects that can be traced from a GCRoot will get their mark bit set.
- The Stage is a GCRoot. Loaders are GCRoots.

Mark and Sweep (Cont.)

- **Sweep Phase:**
  - During sweep phase all objects that are not marked are garbage collected.
  - Weak references are not marked and hence collected when its not in use.
- Mark and Sweep is incremental.



What does all this mean to us?

- To run GC create more objects. 
- Avoid creating objects in loops.
- If possible pool similar objects. Greatest example is ItemRenderers.
- Mark is the most intensive part of GC so create objects judiciously.
- Avoid deep nesting objects. Smaller GCRoot tracing.
- All applications in MORCOM run in single Flash player instance. It impacts all other applications when memory leak or performance degrades.

Where to go from here?

- Events & Event Propagation Model
- Flash Player Security – Sandboxing Fundamentals
- Flash Co-ordinate System – Content, Global & Local
- Adobe AIR – Integrated Runtime
- Improving Performance of Flex Applications
- Flash XML Sockets
- Flash Graphics & Drawing
- Internals of the AVM2

Q & A

OMG! No matter how hard I think, I still have questions.

The End