存储库样本

*此示例与2015年11月的Xbox One XDK或更高版本兼容*

# 描述

此样本演示了几种可用于管理存储库的方法。

* 随机 - 通过 VirtualAlloc 分配的基线内存块。
* 固定 - 演示如何在特定内存地址分配内存。
* 只读 - 演示如何将内存转换为只读。
* 存储库切换 - 演示如何创建可旋转的存储库链。
* 共享 - 演示如何创建共享相同物理位置的多个存储库。

# 使用样本

按下控制器上的相应按键以展示每次演示。将报告成功或失败。

# 实施说明

## 随机

此演示将作为其他演示的基线。使用 VirtualAlloc 分配一个内存块，然后从磁盘读取二叉树到内存块中。需要修复磁盘中的数据。二叉树在内部将索引存储到所分配的内存块中。所有二叉树的内部指针都需要根据索引进行更新，因为基础地址在程序运行之间会发生变化。

## 固定

此演示与随机地址演示的运行方式相同。主要区别在于基础地址存储在数据文件中。这使得程序在每次运行中可以在相同的存储位置重新创建相同区块。这意味着无需内部索引，也不需要指针修复。可以更快地从磁盘加载数据，几乎没有 CPU 开销。

## 存储库切换

此演示展示了如何创建可进行切换的多个存储库。这会将虚拟地址交换为两个物理内存块。可使用两个以上的存储库，然后在这些存储库之间旋转。这样的几种用途为创建记录数据、用于保存回放的帧数据，以及在缓冲区之间移动内存的其他位置。这种方法允许移除存储器移动操作，并会产生性能得到明显改善。

## 共享

此演示展示了如何创建所有指向同一物理块的多个内存块。如果创建两个彼此相邻的虚拟块，则允许环形缓冲区无需对边界副本进行多个内存复制操作。数据跨越一个虚拟块末端写入时，将自动换行至物理块开端。

另一个用途是创建具有不同权限的多个虚拟指针，例如读写和只读。但是必须注意混合调整缓存使用的某些类型的权限，例如合并写和可缓冲。这将导致 CPU 在访问一个指针或另一个可能难以追踪错误的指针时，以不同方式缓存数据。

## 只读

此演示展示了如何创建只读页面。使用共享演示执行此操作，其中一个区块标记为读写，另一个区块标记为只读。这种模式的主要用途在于追踪随机内存损坏。例如，标题所使用的静态数据。文件加载系统可使用读写地址来创建数据。标题的其余部分使用只读指针进行访问。如果存在内存损坏问题，将在源代码中导致问题的确切位置引发异常。

# 更新历史记录

2016年10月初次发布