메모리 뱅크 샘플

*이 샘플은 Xbox One XDK 2015 년 11 월 이상 버전과 호환됩니다.*

# 설명

이 샘플은 메모리 뱅크 관리에 사용할 수 있는 여러 가지 방법을 보여줍니다.

* 랜덤 - VirtualAlloc을 통해 할당된 기본 메모리 블록입니다.
* 고정 - 특정 메모리 주소에 메모리를 할당하는 방법을 보여줍니다.
* 읽기 전용 - 메모리를 읽기 전용으로 변환하는 방법을 보여줍니다.
* 뱅크 전환 - 순환할 수 있는 메모리 뱅크의 체인을 만드는 방법을 보여줍니다.
* 공유 - 동일한 물리적 위치를 공유하는 여러 메모리 뱅크를 만드는 방법을 보여줍니다.

# 샘플 사용하기

각 데모마다 컨트롤러의 해당 버튼을 누릅니다. 그러면 성공 또는 실패를 보고합니다.

# 구현 정보

## 랜덤

이 데모는 다른 데모의 기준 역할을 합니다. VirtualAlloc을 사용하여 메모리 블록을 할당한 다음 디스크의 이진 트리를 메모리 블록으로 읽습니다. 디스크의 데이터는 고정되어야 합니다. 내부적으로 이진 트리는 할당된 메모리 블록에 인덱스를 저장합니다. 이진 트리에 대한 모든 내부 포인터는 프로그램의 실행 사이에 기본 주소가 변경되기 때문에 인덱스를 기반으로 업데이트해야 합니다.

## 고정

이 데모는 랜덤 주소 데모와 동일한 방식으로 작동합니다. 주요 차이점은 기본 주소가 데이터 파일에 저장된다는 것입니다. 이렇게 하면 프로그램이 실행될 때마다 동일한 메모리 위치에서 동일한 블록을 재생성할 수 있습니다. 이것은 내부 인덱스가 필요하지 않으며 포인터 픽스업이 필요 없다는 것을 의미합니다. CPU 오버헤드가 거의 0인 디스크에서 데이터를 훨씬 더 빨리 로드할 수 있습니다.

## 뱅크 전환

이 데모에서는 여러 메모리 뱅크를 전환하는 방법을 보여줍니다. 그러면 가상 주소가 두 개의 물리적 메모리 블록으로 바뀝니다. 두 개 이상의 메모리 뱅크를 사용하고 서로 순환할 수 있습니다. 이를 위한 여러 가지 용도는 로깅 데이터 생성에서, 재생 저장에 사용되는 프레임 데이터 및 다른 위치 메모리가 버퍼 간에 이동되는 것입니다. 이를 통해 메모리 이동 작업이 제거되고 성능이 크게 향상됩니다.

## 공유

이 데모에서는 모두 동일한 물리적 블록을 가리키는 여러 메모리 블록을 만드는 방법을 보여줍니다. 두 개의 가상 블록이 서로 인접하여 생성되면 경계 복사본에 대해 다중 메모리 복사 작업을 필요로 하지 않는 링 버퍼를 사용할 수 있습니다. 데이터가 하나의 가상 블록의 끝에 기록되면 물리적 블록의 시작 부분에 자동으로 래핑됩니다.

또 다른 용도는 읽기-쓰기와 읽기-전용과 같이 다른 권한을 가진 다중 가상 포인터를 생성하는 것입니다. 캐시 사용을 조정하는 특정 유형의 사용 권한 (예: 쓰기 결합 및 캐시 가능)을 혼합할 때는 주의를 기울여야 합니다. 이것은 하나의 포인터에 접근할 때 CPU가 다른 방식으로 데이터를 캐싱하게 만들거나 버그 추적을 어렵게 만들 수 있습니다.

## 읽기 전용

이 데모에서는 읽기 전용 페이지를 만드는 방법을 보여줍니다. 한 블록이 읽기-쓰기 상태로 표시되고 다른 블록이 읽기 전용으로 표시되는 공유 데모를 사용하여 실현합니다. 이 패턴의 주요 용도는 랜덤 메모리 손상을 추적하는 것입니다. 예를 들어, 제목에 사용되는 정적 데이터를 들 수 있습니다. 파일 로딩 시스템은 데이터를 생성하기 위해 읽기-쓰기 주소를 사용할 수 있습니다. 나머지 제목은 액세스를 위해 읽기 전용 포인터를 사용합니다. 메모리 손상 문제가 있는 경우 문제를 일으키는 소스의 정확한 위치에서 예외가 발생합니다.

# 업데이트 기록

초기 출시 2016년 10월