

심볼 프록시 클라이언트 샘플

*이 샘플은 Xbox One XDK 2015 년 6 월 버전과 호환됩니다.*

*호환성 참고 사항: 6 월 XDK 업데이트에는 xbSymbolProxy가 소스 라인 정보를 제대로 분석하지 못하는 알려진 문제점이 있습니다. 이것은 8월 XDK에서 수정되었으며 6 월 XDK QFE 1에 포함될 것입니다.*

# 설명

이 샘플은 SymbolResolve API를 사용하여 콘솔의 주소에 대한 심볼 정보를 확인하는 방법을 보여줍니다. SymbolResolve.h, GetSymbolFromAddress() 및 GetSourceLineFromAddress ()에는 현재 두 개의 API가 있습니다. 이러한 API는 예를 들어 캡처한 백 추적에서 반환 주소를 디코딩하는 경우와 같은 콘솔 측 진단 코드에서 유용합니다. 자세한 내용은 XDK 설명서를 참조하세요.

이 샘플에서는 콜스택 캡처 및 디코딩을 위한 CallStack.h/.cpp의 유용한 유틸리티 코드도 제공합니다. 백 트레이스를 캡처하는 세 가지 헬퍼 함수:

현재 스레드에 대한 백 트레이스 캡처

CaptureBackTraceFromCurrentThread()

제공된 컨텍스트 레코드로 시작하는 백 트레이스를 캡처

CaptureBackTraceFromContext()

제공된 컨텍스트 레코드로 시작하는 콜스택의 스택 프레임 수를 셉니다.

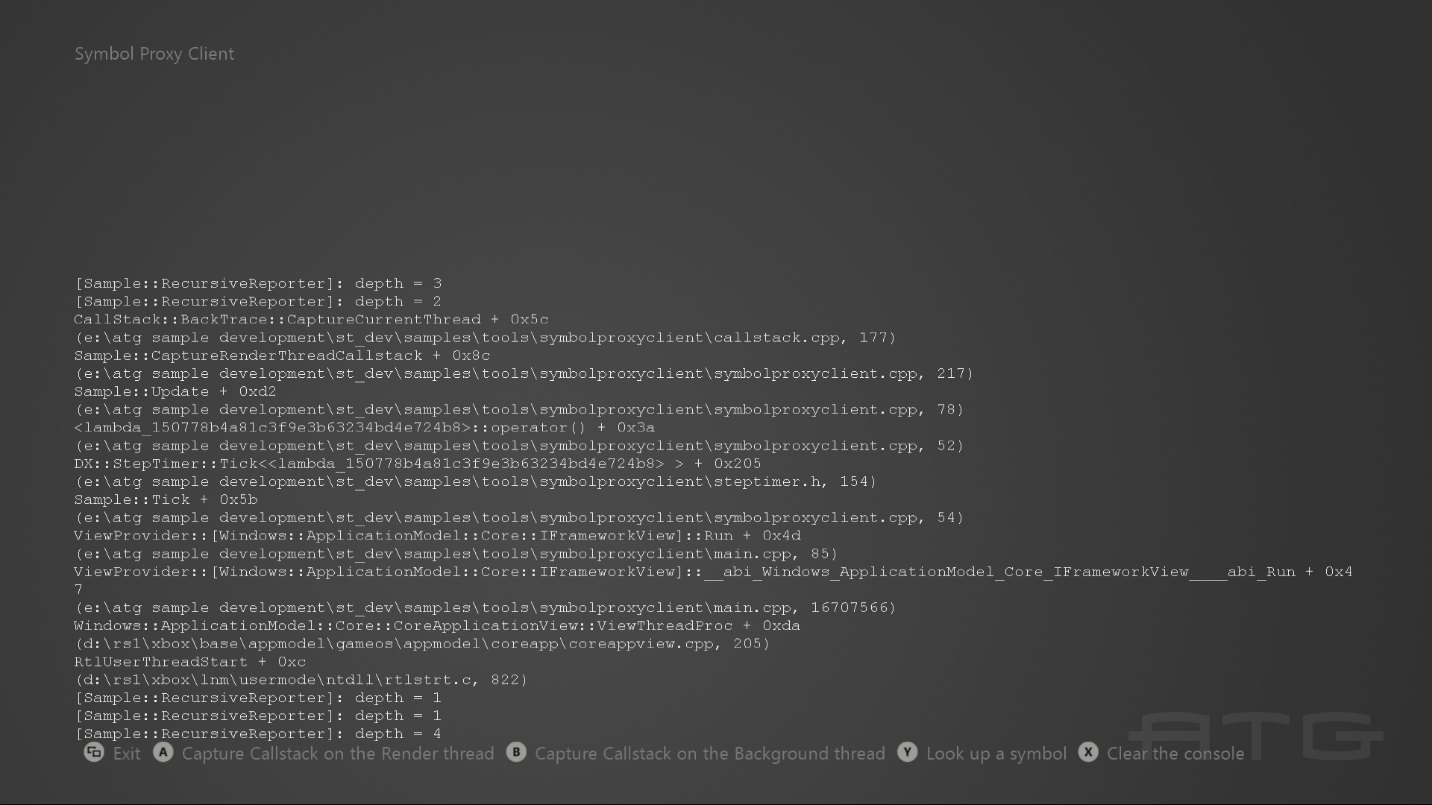
GetFrameCountFromContext()

또한 CallStack.h/.cpp의 BackTrace 클래스는 백 트레이스를 캡처하고 반환 주소에 대한 심볼을 확인하는 프로세스를 단순화합니다.

# 샘플 사용하기

|  |  |
| --- | --- |
| 액션 | 게임 패드 |
| 종료 | 보기 버튼 |
| 렌더 스레드에서 콜 스택 캡처 | A 버튼 |
| 백그라운드 스레드에서 콜 스택 캡처 | B 버튼 |
| 심볼을 찾기 | Y 버튼 |
| 텍스트 콘솔 지우기 | X 버튼 |

심볼을 해결하려면 xbConnect.exe를 사용하여 콘솔에 연결하도록 구성된 개발 시스템에서 xbSymbolProxy.exe를 실행해야 합니다. 그렇지 않으면 샘플은 심볼을 해석할 수 없을 때 단순히 오류를 출력합니다.



# 구현 정보

**심볼과 성능 해결**

심볼을 해결하려면 API가 데스크톱 컴퓨터에서 실행중인 xbSymbolProxy.exe와 통신해야 합니다. 이 작업은 느린 작업이므로 성능이 중요한 스레드가 아닌 백그라운드 스레드에서 비동기 적으로 수행해야 합니다 (다른 느린 작업과 마찬가지로). 이 모범 사례를 보여주기 위해 샘플은 모든 심볼 분석을 비동기적으로 수행합니다.

BackTrace 클래스는 캡처 및 해결 단계를 두개의 다른 작업으로 나눕니다. BackTrace::CaptureCrossThread 및 BackTrace::CaptureCurrentThread를 사용하여 백 트레이스를 캡처합니다. 이는 상당히 효과적이며 백 트레이스를 캡처하는 스레드에서 성능 비용을 피할 수 없습니다. BackTrace::Resolve ()를 사용하여 심볼 프록시와 통신하고 주소에 대한 심볼 정보를 확인하세요. 이것은 매우 느릴 수 있지만 백 트레이스를 캡처한 동일한 스레드에서 심볼을 해결할 필요가 없으므로 백그라운드 스레드에서 BackTrace::Resolve ()를 비동기적으로 호출할 수 있습니다.

**스레드 일시 중단**

백 트레이스를 캡처하는 동안 스택 프레임을 추가하거나 제거하면 안됩니다. 현재 스레드에서 백 트레이스를 캡처하는 경우 쉽게 달성할 수 있습니다. 그러나 다른 스레드에서 백 트레이스를 캡처하는 경우 해당 메커니즘을 사용하여 해당 스레드를 "고정"해야 합니다. BackTrace 클래스는 백 트레이스를 캡처하기 전과 후에 SuspendThread와 ResumeThread를 간접적으로 호출합니다. SuspendThread와 ResumeThread에 대한 호출은 ThreadSuspender라는 헬퍼 클래스에서 관리됩니다.

스레드가 일시 중단된 경우 임계 섹션, 뮤텍스 또는 세마포어와 같은 일종의 동기화 프리미티브를 소유하고 있을 가능성이 있음을 이해하는것이 중요합니다. 이로 인해 일시 중단된 스레드를 다시 시작하기 전에 일시 중단 스레드가 동기화 프리미티브의 소유권을 가져와야 하는 경우 교착 상태 시나리오가 발생할 수 있습니다. 이것이 SuspendThread를 사용하는것이 일반적으로 좋지 않은 이유입니다.

다른 스레드가 일시 중단된동안 일시 중단 스레드가 메모리를 할당해야 하는 경우 특히 문제가 되는 경우가 있습니다. 일시 중단된 스레드가 힙에서 잠금을 보유한 경우 교착 상태가 발생합니다. 이러한 교착 상태를 피하기 위해 BackTrace 클래스는 백 트레이스의 반환 주소를 저장하기 위해 전역 정적 버퍼를 사용하므로 백 트레이스를 캡처하는 동안 메모리를 할당할 필요가 없습니다. 클래스는 64 스택 프레임을 위한 충분한 공간을 확보합니다.