달빅 DEX 파일 브라우저의 설계 및 구현

소경영*, 정택희*, 박종필**, 고광만**
*전북대학교 소프트웨어공학과
**상지대학교 컴퓨터정보공학부
e-mail: {kyso, wjdxorgml}@jbnu.ac.kr, {jppark, kkman}@compiler.sangji.ac.kr

Design and Implementation of Dalvik DEX File Browser

Kyung-Young So*, Taek-Hee Jung*
*Dept of Software Engneering, Chonbuk National University
Jong-Pil Park*, Kwang-Man Ko**
**School of Computer and Information Engineering, Sang-Ji University

요 약

안드로이드 플랫폼에 적합한 어플리케이션 보급이 급증하면서 달빅(dalvik)에 관련된 다양한 연구 시도가 진행되고 있다. 특히, Java 클래스 파일로부터 dx에 의해 생성되는 DEX 파일의 구조 및 상세 정보를 시각적으로 분석하고 이를 응용하기 위한 노력은 다양한 성능 향상의 효과를 기대할 수 있다. 이 논문에서는 달빅 가상머신의 실행 파일인 DEX 파일의 구조 및 정보를 세분화하여 시각적으로 쉽게 접근할 수 있는 브라우저를 설계하고 구현한다. 이를 통해서, DEX 파일의 구조 및 정보를 보다 쉽게 접근하고 이용할 수 있으며 디어셈블러(smali/baksmali) 편리하게 사용할 수 있도록 하였다.

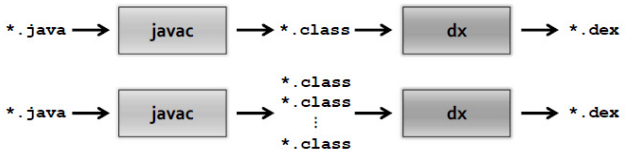
1. 서론

최근 안드로이드 어플리케이션의 보급이 급증하면서 실행 성능 향상, 사용 편리성, 개발 용이성 등에 관련된 다양한 연구시도와 상용화 제품이 출시되고 있다. 안드로이드 어플리케이션은 안드로이드 가상머신인 달빅(dalvik) 인스턴스와 함께 자신의 프로세스에서 실행되며 달빅은 여러 인스턴스가 효율적으로 실행될 수 있도록 구현되어 있다. 달빅에서 Java 어플리케이션(*.java)을 실행하기 위해서는 Java 컴파일러(javac)에 의해 생성된 클래스 파일(*.class)을 최소화된 메모리 영역을 차지하는 최적화된 Dalvik Executable 형식의 파일(*.dex)로 변환해야 한다. 달빅은 레지스터 기반의 가상머신으로서 dx 툴을 이용하여 Java 클래스 파일을 dex 형식으로 변환한다[1,2].

바이너리 형태로 구성된 달빅 DEX 파일은 달빅에서 올바른 수행을 위한 많은 정보를 가지고 있으며 실질적으로 클래스 파일의 구조 및 정보를 추출하기 위한 도구들이 사용되고 있다. 하지만 이러한 도구들은 대부분 텍스트 형식으로 구성되어 DEX 파일의 내용을 접근하는데 많은 불편함을 가지고 있다. 또한 DEX 파일의 핵심 부분인 달빅 바이트코드를 이해하기 위해서는 복잡한 클래스 파일에 대한 접근이 제한되고 있다[3,4]. 이 논문에서는 달빅 가상머신의 실행 파일인 DEX 파일의 구조 및 정보를 세분화하여 시각적으로 쉽게 접근할 수 있는 브라우저를 설계하고 구현한다. 이를 위해, Java 어플리케이션의 클래스 파일(*.class)을 입력으로 받아 헤더, 상수 풀, 데이터 부분으로 구성된 DEX 파일(*.dex)의 상세 구조를 시각화된 윈도우에 표현한다. 이러한 연구는 클래스 파일로부터 생성되는 DEX 파일의 매핑 관계를 시각적으로 확인할 수 있다. 둘째, DEX 파일의 구조를 보다 상세하게 파악할 수 있으며 디어셈블리(smali) 과정에서 편리하게 활용할 수 있다[5].

2. DEX 파일 구조 및 클래스

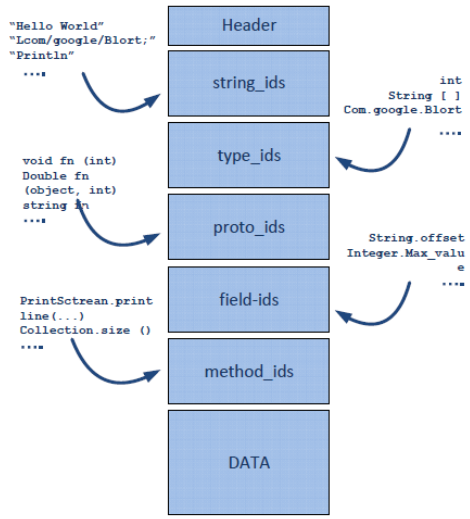
달빅에서 DEX 파일을 실행하기 위해서는 [그림 1]과 같이 Java 컴파일러로부터 생성된 *.class 파일[7]들을 안드로이드에서 제공하는 dx 도구를 사용하여 *.dex 파일로 변환시킨다. *.class 파일이 여러 개가 존재하더라도 하나의 .dex 파일로 통합되고, 상수 풀을 공유하게 된다. 이렇게 통합된 *.dex 파일은 일반적인 JAR 파일과 비교했을 때 절반 이상으로 파일 크기가 감소된다.



[그림 1] dex 파일 변환 과정

dex 파일은 [그림 2]와 같이 여러 영역으로 구성되어 있으며 각 영역은 특정 타입으로, “string_ids” 영역은 dex 파일에 포함된 모든 문자열을 나타낸다. dex 파일의 시작 부분에는 헤더가 포함되어 있고

헤더에는 Magic 넘버와 체크섬을 포함하고 있으며 string table, class list, field table, method table 등의 오프셋 정보(절대값)를 가지고 있다.

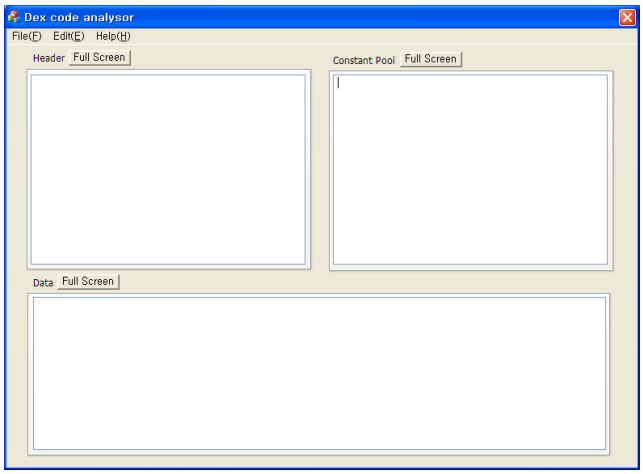


[그림 2] DEX 파일 구조

Java 어플리케이션은 메타 데이터 파일을 포함하는 *.jar 또는 *.apk 파일 형식이다. classes.dex는 Java 바이트코드를 달빅 바이트코드로 전환시키는 dx 툴이 사용되기 전에 압축이 해제되어야 한다. 다른 재정렬, 최적화, 검증과 같은 작업은 달빅 바이트코드 상에서 수행된다. DEX 파일을 실행시키기 위한 첫 번째 동작은 달빅 캐시 디렉토리에 Dalvik-cache 파일을 생성한다. 캐시에서 파일을 작업, 생성, 삭제, 수정하기 위해서는 적절한 권한이 요구된다. 둘째, classes.dex의 압축을 해제한다. 이 과정에서 시스템에 빠르고 쉬운 접근을 위해 바이트코드 스와핑과 구조 재정렬을 포함한 메모리 매핑이 수행된다. 체크는 데이터 인덱스와 파일 오프셋이 검증 범위 안에서 이루어지는 것을 보장하기 위해 만들어진다. 모든 클래스는 가상머신에서 로딩하고 실행함으로써 검증과 최적화를 수행한다. 검증 프로세스동안, 특정한 것은 리소스 락을 유발할 수 있는 검증과 최적화 실패를 할 수 있다. 따라서 이 프로세스는 어플리케이션이 실행되는 가상머신 아닌 독립된 가상머신에서 수행된다. 실행시간 전에 검증과정을 포함하는 불법 명령어의 확인이 요구되며 DEX 파일에서 모든 클래스에 의해 정의된 메소드의 모든 명령어를 스캔한다. 최적화에서 가상머신 인터프리터는 내부 데이터 구조에 대한 포인터를 갖는 상수 풀 참조를 대체함으로써 사용되고 압축된 코드의 조각을 최적화한다. 따라서 항상 작동하는 작업은 더 간단한 형태로 대체된다[6].

2. DEX 파일 브라우저 설계 및 구현

이 논문에서는 DEX 파일의 구조 및 정보를 시각적으로 표현하기 위해 *.class 파일을 입력으로 받아 [그림 3]과 같이 DEX 파일의 전체 구조를 헤더 부분, 상수 풀 부분, 데이터 부분으로 구분하여 소스 내용을 브라우저에 표시한다.



[그림 3] DEX 파일 브라우저 메인화면

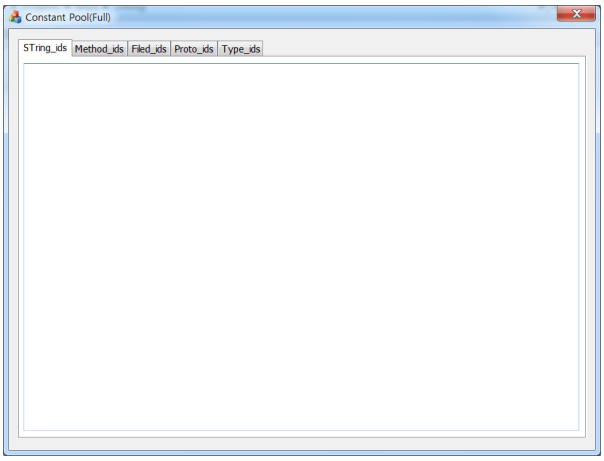
DEX 파일의 시작 부분에는 헤더가 포함되어 있고 헤더의 내용은 [그림 4]와 같이 Magic 넘버와 체크섬을 포함하고 있고 string table, class list, field table, method table 등의 오프셋 정보(절대값)를 가지고 있다.

Offset	Size	Description
0x0	8	'Magic' value: "dex\n009 \0"
0x8	4	Checksum
0xC	20	SHA-1 Signature
0x20	4	Length of file in bytes
0x24	4	Length of header in bytes (currently always 0x5C)
0x28	8	Padding (reserved for future use?)
0x30	4	Number of strings in the string table
0x34	4	Absolute offset of the string table
0x38	4	Not sure. String related
0x3C	4	Number of classes in the class list
0x40	4	Absolute offset of the class list
0x44	4	Number of fields in the field table
0x48	4	Absolute offset of the field table
0x4C	4	Number of methods in the method table
0x50	4	Absolute offset of the method table
0x54	4	Number of class definitions in the class definition table
0x58	4	Absolute offset of the class definition table

[그림 4] DEX 파일 헤더

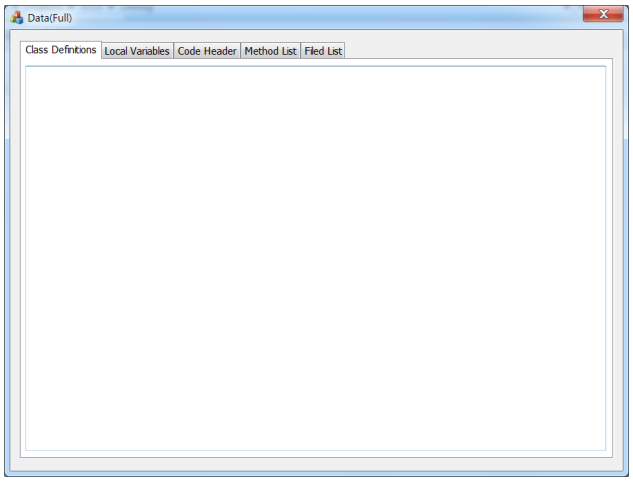
상수 풀에 표현되는 정보는 apk를 형성하는 클래스 파일의 상수 풀의 정보가 하나의 상수 풀에 스트링 상수 풀, 타입 명칭 상수 풀, 프로토 명칭 상수 풀, 필드 명칭 상수 풀, 메소드 명칭 상수 풀로 세부

분화 되어 [그림 5]와 같이 표시된다. 세부적인 상수 풀의 정보를 표시하기 위해서는 각 상수 풀 명칭의 탭을 클릭한다.



[그림 5] 상수 풀 정보 표현을 위한 DEX 브라우저

데이터 부분은 클래스 정의, 필드 리스트, 메소드 리스트, 코드 헤더와 지역 변수 정보가 [그림 6]과 같이 표현된다.



[그림 6] 데이터 정보 표현을 위한 DEX 브라우저

3. 결론 및 향후 연구

최근 안드로이드 플랫폼에서 다양한 어플리케이션이 개발되면서 안드로이드 플랫폼에 대한 다양한 분석 연구가 진행되고 있다. 달빅 가상머신에서 실행되는 DEX 파일은 Java 클래스 파일로부터 생성되는 정보를 효율적으로 저장하고 있으므로 DEX 파일의 구조 및 정보를 분석하기 위한 도구에 대한 개발 연구가 요구된다. 이 논문에서는 달빅 가상머신의 실행 파일인 DEX 파일의 구조 및 정보를 세분화하여 시각적으로 쉽게 접근할 수 있는 브라우저를 설계하고 구현한다. 이를 위해, Java 어플리케이션의 클래스 파일(*.class)을 입력으로 받아 헤더, 상수 풀, 데이터 부분으로 구성된 DEX 파일(*.dex)의 상세 구조를 시각화된 윈도우에 표현한다. 앞으로, 현재 개발되고 있는 DEX 파일 브라우저에 대한 보다 완성도를 높이고 보다 세부적인 정보를 표현할 수 있는 연구를 진행할 예정이다.

참고문헌

[1] Dalvik Virtual Machine, http://www.dalvikvm.com/.

[2] Dan Bornstein. Dalvik virtual machine: Internals. http://sites.google.com/site/io/dalvik-vm-internals/2008-05-29-Presentation-Of-Dalvik-VM-Internals.pdf.

[3] Koji Hisano. A clean room implementation of android's dalvik virtual machine on java. http://code.google.com/p/android-dalvik-vm-on-java/.

[4] Peter Haggar. Understanding java bytecode. http://www.ibm.com/developerworks/ibm/library/it-haggar_bytecode/.

[5] Y. Shi, K. Casey, M.A. Ertl, and D. Gregg. Virtual machine showdown: stack versus registers. ACM Transactions on Architecture and Code Optimization (TACO), 4(4):2, 2008.

[6] J.E. Smith and R. Nair. Virtual machines: versatile platforms for systems and processes. Morgan Kaufmann Pub, 2005.

[7] T. Suganuma, T. Ogasawara, M. Takeuchi, T. Yasue, M. Kawahito, K. Ishizaki, H. Komatsu, and T. Nakatani. Overview of the IBM Java just-in-time compiler. IBM Systems Journal, 39(1):175193, 2000.