Android C++ Native SDK

Android NDK - Android Native Development Kit

W3L AG
info@W3L.de

2014

Inhaltsverzeichnis

zurück

Android

Übersicht

- **Software-Plattform für mobile Geräte**
- **Entwickelt von Open Handset Alliance**
- **Programmiersprache Java**
- **SDK für Windows, Linux und Mac OS X erhältlich**
- **Java Development Kit erforderlich**
- **Entwicklungsumgebung**
  - Eclipse mit ADT-Plugin
  - Android Studio

Android

Laufzeitumgebung

- **Multi-User Linux System**
  - Laufende Applikationen erhalten vom System eine eindeutige User-ID
  - Zugriffsrechte für alle Dateien der Applikation werden nur der zugeteilten User-ID bereitgestellt
- **Applikationen werden in eigenständigen System-Prozessen ausgeführt**
- **Principle of least privilege**
- **Dalvik VM bildet Laufzeitumgebung für Applikation**

Android

Applikation

- **Activity**
  - View für Benutzer-Interaktion
- **Service**
  - Komponente für lang laufende Prozesse im Hintergrund
  - Unterstützt keine UI
- **Content Provider**
  - Verwaltet Zugriff auf Daten
  - Zugriff auf Datenbanken oder Dateisystem
- **Broadcast Receiver**
  - Registrierung auf Systemereignisse

Android NDK

Einleitung

- **Ermöglicht die Entwicklung von native Applikationen**
- **Programmiersprache C und C++**
- **Geeignet für CPU-Intensive Prozesse, die wenig Speicher benötigen**
  - Audio, Bild und Videobearbeitung
- **Systemvoraussetzungen**
  - Linux, OS X oder Windows
  - Cygwin 1.7 für Windows
- **Zielsystem**
  - Android ab Version 1.5
  - Prozessorarchitektur
    - ARMv5 und ARMv7
    - MIPS
    - x86

Unterstützte Bibliotheken

C und C++

- **Minimale C++ Unterstützung**
- **RTTI wird nicht unterstützt**
- **C++ Exceptions werden nicht unterstützt**
- **Alternative Runtime kann eingebunden werden**
  - gabi++
  - stlport
  - gnustl
  - libc++

Unterstützte Bibliotheken

Hilfsfunktionen

- **Math Library**
  - Mathematische Funktionen und Makros
  - Trigonometrische Funktion
- **Android Log-Support**
  - Ausgabe von Log-Informationen in Log-Buffer
  - Priorität kann festgelegt werden
  - Log-Buffer sehr klein (< 64 KB)
  - Text kann im Logcat ausgelesen werden
- **Zlib**
  - Komprimieren und Dekomprimieren von Daten
  - Deflate-Algorithmus
- **Dynamic Linker Lib**
  - Dynamisches nachladen von Programmbibliotheken

Unterstützte Bibliotheken

Multimedia

- **OpenGL ES**
  - Bibliotheken zur Ansteuerung von Grafikhardware für eingebettete Systeme
  - Darstellung von 3D-Szenen
  - Datentypen sind beschränkt
- **OpenSL ES**
  - Bibliotheken für die native Audioverarbeitung für eingebettete Systeme
  - Spezifische Erweiterungen für Android
- **OpenMAX AL**
  - Bibliotheken für die Audio- und Videoverarbeitung
- **EGL**
  - Programmierschnittstelle zwischen OpenGL ES und Fenster-Management

Unterstützte Bibliotheken

Android

- **JNIGraphics**
  - Ermöglicht Zugriff auf Pixel-Buffer von Java-Bitmap Objekten
  - Bildinformationen werden bereitgestellt

- **Android Native Application API**
  - Native Activity
  - Registrierung auf Eingaben und Sensoren
  - Fenstermanagement
  - Zugriff auf Assets oder OBB-Dateien (Opaque Binary Blob)

Entwicklung in NDK

Hello World

- **SDK Android Projektstruktur**
- **Projektstruktur mit Verzeichnis `/jni` erweitern**
- **Projektbeschreibung `Application.mk`**
  - Zielprozessor
  - Einstellungen für Compiler und Linker
- **Projektbeschreibung `Android.mk`**
  - Module
  - Quelltext-Dateien

Java Native Interface

- **Schnittstelle für Zugriff auf plattformspezifische Funktionen**
- **Native Funktionen können zur Laufzeit von Java-Code aufgerufen werden**
- **Bidirektionale Aufrufe möglich**

```
static
{
   System.loadLibrary("example-jni");
}

public void onCreate(Bundle savedInstanceState)
{
   super.onCreate(savedInstanceState);

   TextView tv = new TextView(this);
   tv.setText(jniFunction());
   setContentView(tv);
}

public native String jniFunction();
```

Native Activity

- **Implementierung einer Activity in C-Code**
  - `native_activity.h`
  - `android_native_app_glue.h`
- **Keine UI-Komponenten vorhanden**
- **Applikation weiterhin an eigene VM-Prozess gebunden**
- **Konfigurationseintrag in `AndroidManifest.xml`**

ARM Neon

Multimedia-Erweiterung

- **Erweiterung der Prozessor-Architektur ARM**
- **SIMD (Single Instruction, Multiple Data)**
  - Eine Rechenoperation wird auf mehrere Datenströme ausgeführt
- **Sinnvoll bei parallelen Operationen in der Bild-, Audio- und Videoverarbeitung**
- **Leistungssteigerung von 60 % bis 150 % bei komplexe Videocodecs**
- **Unterstützte Standards**
  - MPEG-4
  - H.264
  - On2 VP6/7/8
  - Real
  - AVS

Entwicklungswerkzeuge

- **Import Modul**
  - NDK-Module außerhalb der Projektstruktur importieren
  - Konfiguration in `Android.mk`
  - Keine zyklischen Abhängigkeiten erlaubt
- **NDK Build**
  - Kommandozeilentool für die Kompilierung
- **NDK Depends**
  - Abhängigkeiten der Bibliotheken werden aufgelöst
  - Option `--print-java` generiert Java-Code für JNI

Entwicklungswerkzeuge

- **NDK GDB**
  - Erzeugt eine Debug-Session für die native Implementierung
  - Ausführbedingungen
    - Maschinencode muss mit NDK Build erzeugt werden
    - Debug-Funktion muss in der Manifest-Datei von Android aktiviert werden
    - Wird nur für Android Version 2.2 oder höher unterstützt
- **NDK Stack**
  - Erleichtert Auswertung von Log-Informationen
  - Stack Trace aus Logcat kann ausgewertet werden

RenderScript

- **Alternative für native Implementierungen**
- **Script-Sprache für performance-kritische Codeabschnitte**
- **Entwickelt für verschiedene Prozessortypen**
- **Laufzeitumgebung nutzt CPU, GPU oder DSP**
- **Script-Code**
  - Programmiersprache von C99 abgeleitet
  - RS- oder RSH-Dateien, abgelegt im Android-Projekt
  - LLVM (Low Level Virtual Machine) kompiliert Code in Bytecode
  - Bytecode wird zur Laufzeit (Just-In-Time) in Maschinencode umgewandelt

- **Unterstützte Versionen**
  - Android Version 2.2 mit Support Library
  - Android Version 3.0

Fazit

NDK in der Praxis

- **Erhöhte Ausführgeschwindigkeit bei rechenintensive Operationen**

- **Steigert Komplexität der Anwendung**
- **Geringe Portabilität**
- **Erhöhte Anforderungen bei Testumgebung**
- **Debug-Möglichkeiten sind begrenzt**
- **Wartungsaufwand wird erhöht**

Links

- **Android NDK**
  https://developer.android.com/tools/sdk/ndk/index.html
- **Android NDK Dokumentation**
  http://www.kandroid.org/ndk/docs/OVERVIEW.html
- **ARM NEON**
  http://www.arm.com/products/processors/technologies/neon.php
- **Android RenderScript**
  http://developer.android.com/guide/topics/renderscript/index.html

Inhouse-Schulungen

Wir bieten Inhouse-Schulungen und Beratung durch unsere IT-Experten und –Berater.

Schulungsthemen

- Softwarearchitektur (OOD)
- Requirements Engineering (OOA)
- Nebenläufige & verteilte Programmierung

Gerne konzipieren wir auch eine individuelle Schulung zu Ihren Fragestellungen.

Sprechen Sie uns an!
Tel. 0231/61 804-0, info@W3L.de

W3L-Akademie

In Zusammenarbeit mit der Fachhochschule Dortmund bieten wir

zwei Online-Studiengänge

- B.Sc. Web- und Medieninformatik
- B.Sc. Wirtschaftsinformatik

und 7 Weiterbildungen im IT-Bereich an.

Besuchen Sie unsere Akademie!
http://Akademie.W3L.de