Εισαγωγή στο Java Native Interface

Βασιλική Βλάχου

Εθνικό Μετσόβιο Πολυτεχνείο
Τεχνολογία Λογισμικού

Java:

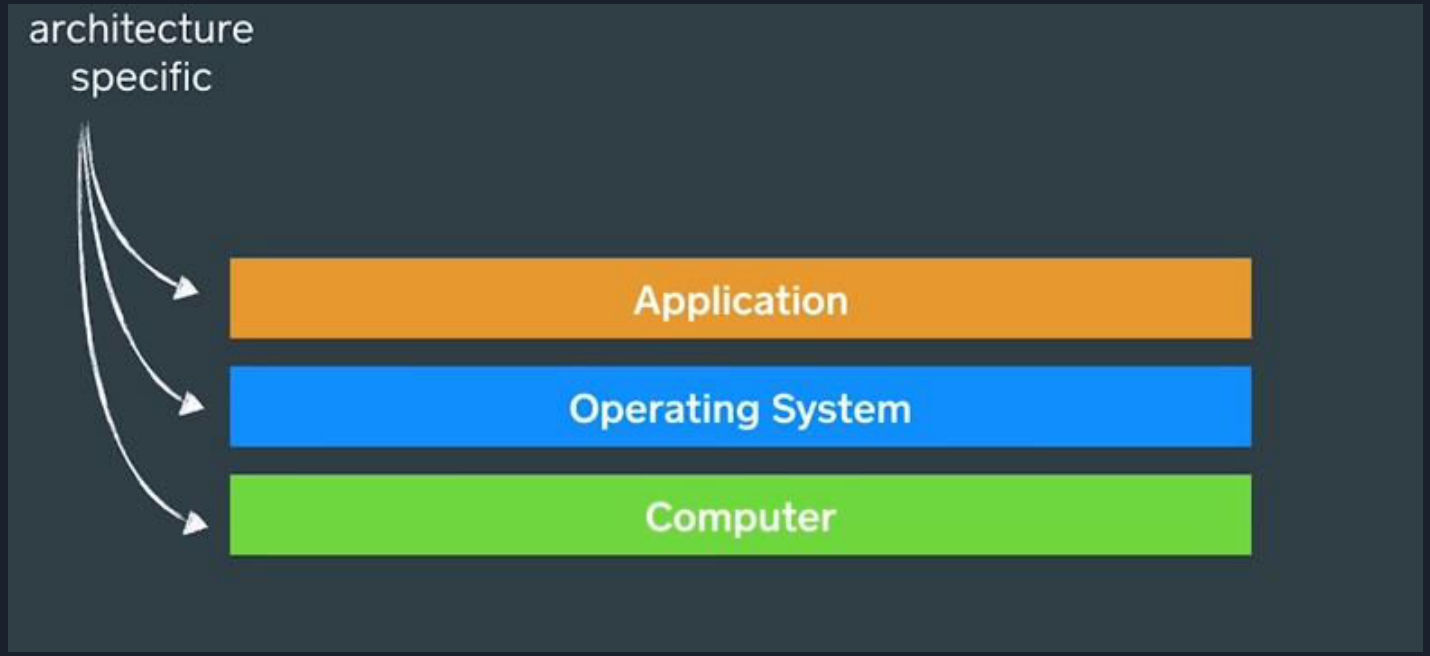
- πολλές βιβλιοθήκες
- αρκετά γρήγορη αν και high level

JVM

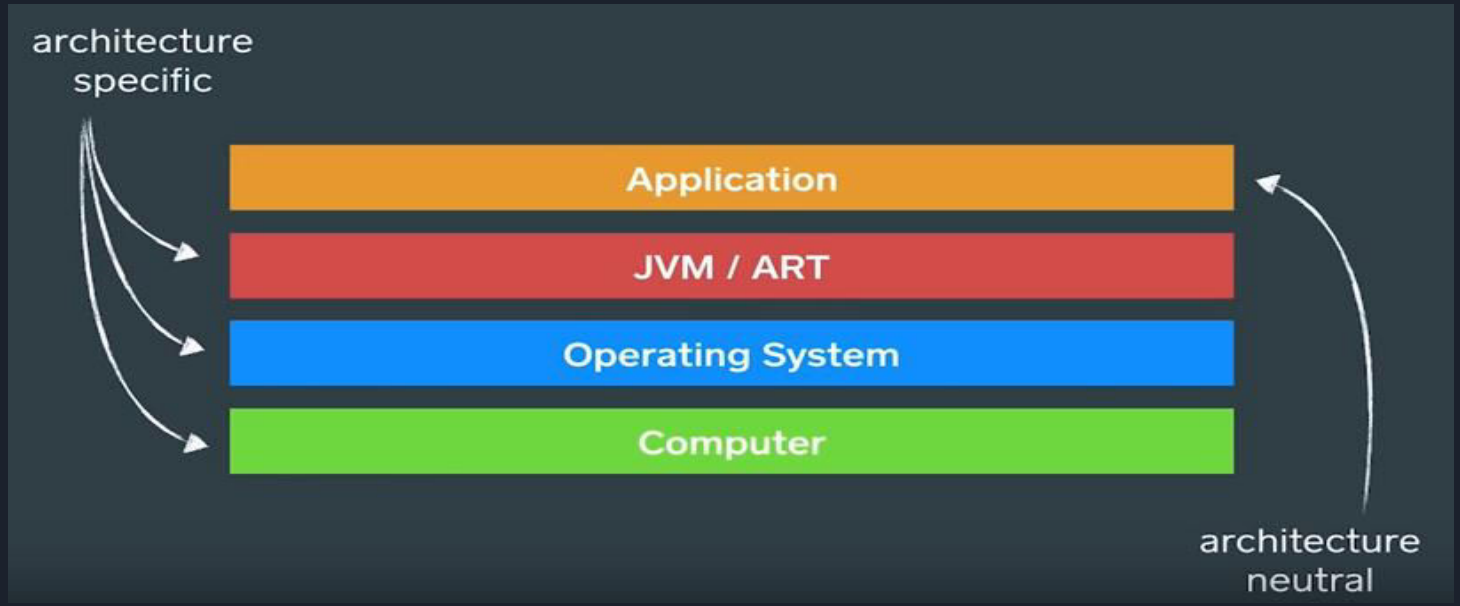
Java **V**irtual **M**achine

- architecture neutral
- garbage collected
- fast

Application architecture πριν τη Java



Application architecture με Java



Τι είναι το JNI

Είναι ένα framework που επιτρέπει σε κώδικα Java που εκτελείτα σε JVM να καλεί και να καλείται από native εφαρμογές και βιβλιοθήκες γραμμένες σε άλλες γλώσσες, όπως c, c++ ή ακόμα και assembly.

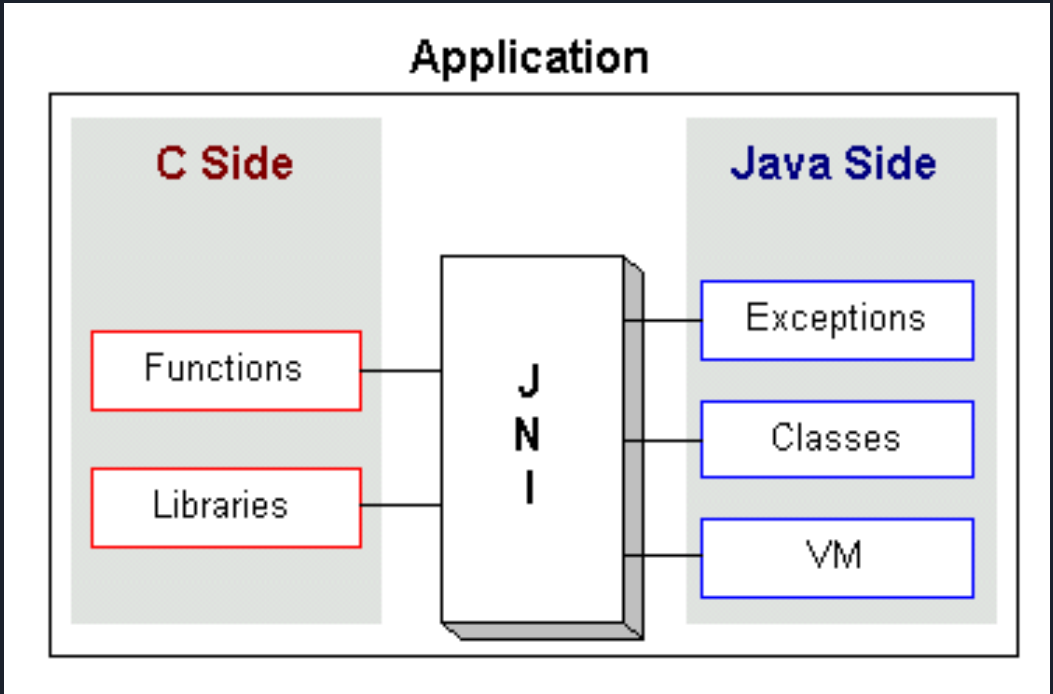
Γιατί να χρησιμοποιήσεις JNI

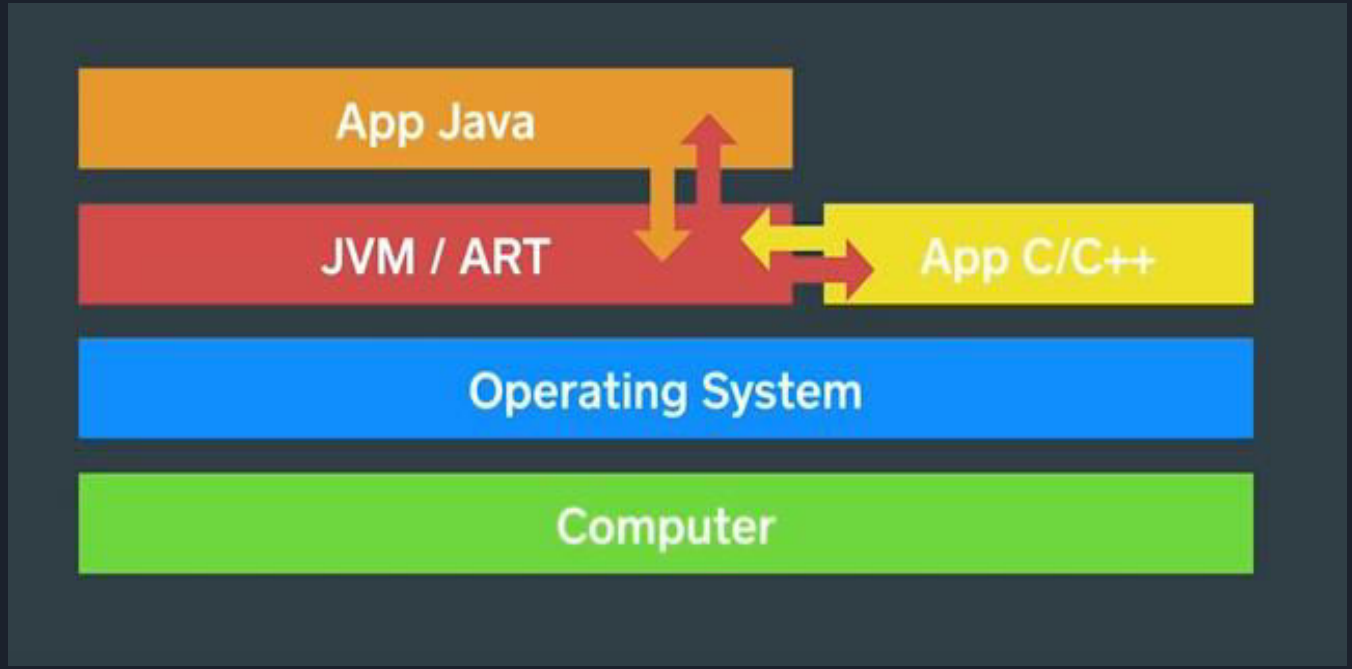
- Ορισμένες λειτουργίες δεν παρέχονται από τη Java
- Χρήση υπάρχοντος κώδικα γραμμένου σε άλλη γλώσσα προγραμματισμού (επαναχρησιμοποίηση κώδικα)
- Υλοποίηση time-critical κώδικα σε γλώσσα χαμηλού επιπέδου (χρήση βέλτιστης γλώσσας για κάθε λειτουργία)

Μειονεκτήματα

- Η εφαρμογή δεν είναι πλέον αρχιτεκτονικά ανεξάρτητη
- Οι κλήσεις native κώδικα προκαλούν μεγάλο performance overhead

Το JNI λειτουγεί σαν “κόλλα” ανάμεσα σε Java και native εφαρμογές.




App Java
JVM / ART
App C/C++
Operating System
Computer

Διαδικασία

- Δημιουργία κλάσης που δηλώνει τη native μέθοδο και περιλαμβάνει τη main που καλεί τη native μέθοδο.
- Compile την κλάση που δηλώνει τις native και main μεθόδους.
- Δημιουργία ενός header file για τη native μέθοδο με javah -jni.
- Υλοποίηση της native μεθόδου στην επιλεγμένη γλώσσα.
- Compile τους header και implementation files σε ένα κοινό library file.
- Τρέξιμο του προγράμματος.

Παράδειγμα “Hello World!”
1) Δημιουργία Κλάσης

```
class HelloWorld {
    public native void displayHelloWorld();

    static {
        System.loadLibrary("hello");
    }

    public static void main(String[] args) {
        new HelloWorld().displayHelloWorld();
    }
}
```

- “native”: ενημερώνει τον compiler ότι η συνάρτηση είναι native language function
- Πρέπει να κάνουμε load τη βιβλιοθήκη που υλοποιεί τη συνάρτηση, ώστε να έχουμε mapping ανάμεσα στην υλοποίηση και τον ορισμό.

2) Compile

Κάνουμε compile την κλάση που δημιουργήσαμε μέσω της εντολής:

javac HelloWorld.java

3) Δημιουργία header file

Δημιουργούμε header file μέσω της εντολής:

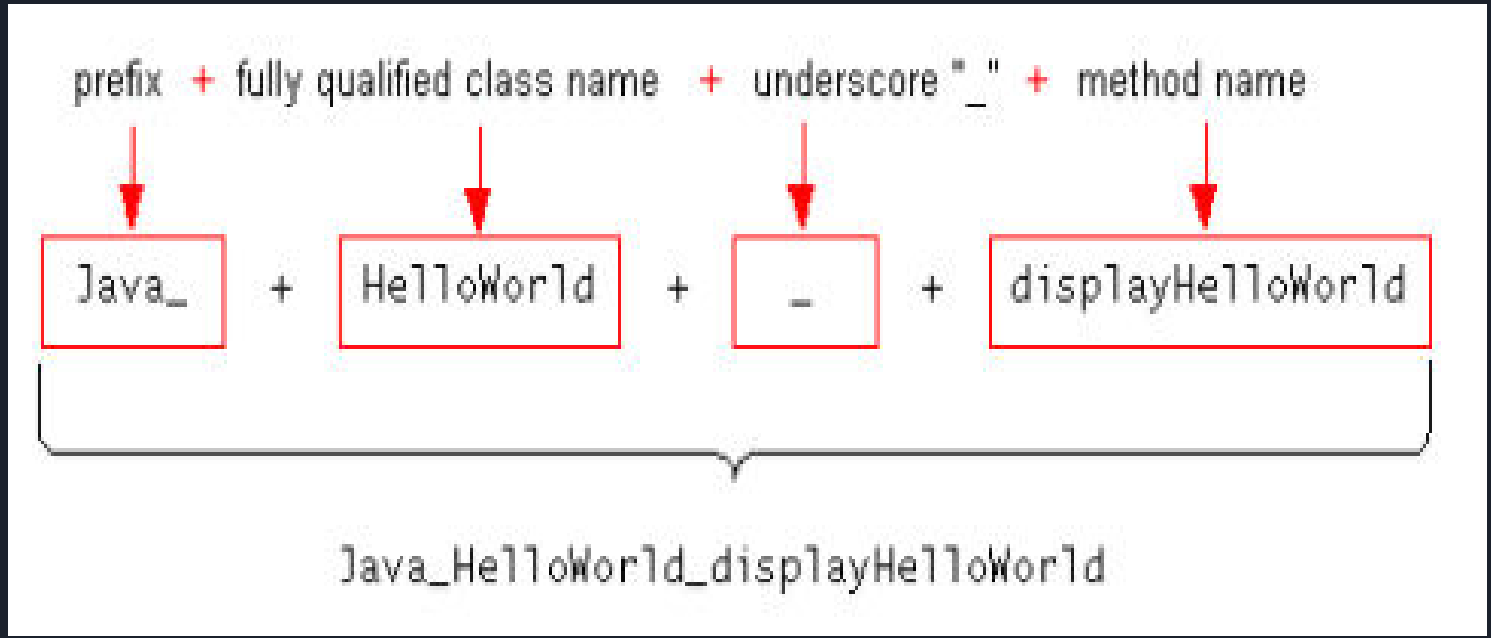
javah -jni HelloWorld

```
/* DO NOT EDIT THIS FILE - it is machine generated */
#include <jni.h>
/* Header for class HelloWorld */

#ifndef _Included_HelloWorld
#define _Included_HelloWorld
#ifdef __cplusplus
extern "C" {
#endif
/*
 * Class:     HelloWorld
 * Method:    displayHelloWorld
 * Signature: ()V
 */
JNIEXPORT void JNICALL Java_HelloWorld_displayHelloWorld
  (JNIEnv *, jobject);

#ifdef __cplusplus
}
#endif
#endif
```

Όνομα της native language function



Παράμετροι

```
JNIEXPORT void JNICALL Java_HelloWorld_displayHelloWorld
  (JNIEnv *, jobject);
```

- JNIEnv: pointer μέσω του οποίου ο native κώδικας έχει πρόσβαση σε παραμέτρους και αντικείμενα που του περνά η εφαρμογή
- jobject: αναφορά στο HelloWorld object

4) Υλοποίηση της native μεθόδου

```c
#include <jni.h>
#include "HelloWorld.h"
#include <stdio.h>

JNIEXPORT void JNICALL
Java_HelloWorld_displayHelloWorld(JNIEnv *env, jobject obj)
{
    printf("Hello world!\n");
    return;
}
```

- Η συνάρτηση πρέπει να έχει την ίδια υπογραφή με αυτή που έχει στο header file.
- jni.h: παρέχει τις πληροφορίες που χρειάζεται ο native κώδικας για καλέσει JNI συναρτήσεις.

5) Δημιουργία Shared Library

- Δημιουργία βιβλιοθήκης που περιέχει τη native μέθοδο.
- Το όνομα της βιβλιοθήκης πρέπει να ταιριάζει με το όνομα βιβλιοθήκης που χρησιμοποιήθηκε στη System.loadLibrary μέθοδο.

6) Τρέξιμο προγράμματος

Τρέχουμε το πρόγραμμα με την εντολή:

java HelloWorld

και παίρνουμε την ακόλουθη έξοδο:

Hello World!

Αλληλεπίδραση με τη Java

Το JNI framework επιτρέπει στη native μέθοδο να χρησιμοποιεί τα Java objects με τον ίδιο τρόπο που τα χρησιμοποιεί κώδικας Java.

Παραδείγματα Λειτουργιών

- Δημιουργία Java Objects
- Χρήση των objects στην υλοποίηση λειτουργιών
- Ενημέρωση objects

Java Strings σε Native Methods

```c
#include <stdio.h>
#include <jni.h>
#include "Prompt.h"

JNIEXPORT jstring JNICALL
Java_Prompt_getLine(JNIEnv *env, jobject obj, jstring prompt)
{
  char buf[128];
  const char *str = (*env)->GetStringUTFChars(env, prompt, 0);
  printf("%s", str);
  (*env)->ReleaseStringUTFChars(env, prompt, str);
  scanf("%s", buf);
  return (*env)->NewStringUTF(env, buf);
}
```

- jstring ≠ char*
- χρειάζεται μετατροπή απο
- Java string σε native string
- ReleaseStringUTFChars: ενημερώνει το VM ότι το string δε χρησιμοποιείται πλέον και μπορεί να ελευθερωθεί

Βιβλιογραφία:

- https://en.wikipedia.org/wiki/Java_Native_Interface
- https://www.softlab.ntua.gr/facilities/documentation/unix/java/tutorial/native1.1/index.html
- https://www.slideshare.net/izacus/jni-java-native-interface

Σας ευχαριστώ!