**3. Внешний интерфейс Java (JNI)**

Java Native Interface (JNI) — стандартный механизм для запуска кода под управлением виртуальной машины Java (JVM), который написан на языках С/С++ или Ассемблере и скомпонован в виде динамических библиотек; позволяет не использовать статическое связывание. Это даёт возможность вызова функции С/С++ из программы на Java, и наоборот. Более ранние интерфейсы, в отличие от JNI, не удовлетворяли условию двоичной совместимости. (Вики)

~~Инкапсуляция — принцип модульности, система разделяется на модули, «внутренности» которых скрыты от остальных.~~

~~Среды управляемого исполнения — экзоскелет вокруг пользовательского кода.~~

~~Цена модульности — накладные расходы при переходе от одного модуля к другому.~~

* Внешний модуль должен использовать JNI
* Копирование для примитивных типов (параметры и результат)
* Обертывание и обратный вызов для объектов (globalRef, localRef)
  + Требует согласования с планировщиком
  + Потери времени при вызове
  + Потери времени при доступе
  + Возможна изоляция управляемых данных
  + Групповые операции с массивами
  + Усложнение внешнего модуля
* Абстрактный неуправляемый массив (DirectBuffer)

Ссылки передаются по значению → неуправляемый код не получит доступ на внутренние ресурсы машины.

Между быстро и безопасно разработчики Java выбрали безопасно.

В настоящий момент JNI разработана как интерфейс с собственными методами написанными только на С или С++. Используя JNI ваши собственные методы могут:

* Создавать, проверять и обновлять Java объекты (включая массивы и типы **String**)
* Вызывать Java методы
* Ловить и выбрасывать исключения
* Загружать классы и получать информацию о классах
* Выполнять проверку типов во время исполнения

Таким образом, практически все, что вы можете делать с классами и объектами в Java вы можете выполнить с собственными методами.

Пример:  public native int showString(String message);