# Item 54: Use native methods judiciously

## Disadvantages of native methods:

- Not save –
  memory corruption errors may occur
- Platform dependent –
  native methods are less portable
- More difficulties with debugging
- Worse performance doing a small amount of work –
  consume some resources for going into and out of native code

## Use as little native code as possible

## Strive to write good programs rather than fast ones

Don’t sacrifice sound architectural principles for performance

## Strive to avoid design decisions that limit performance

Components interacting between modules and with the outside world are difficult to change.

## Consider the performance consequences of your API design decisions

- Mutable public class – may require a lot of needles defensive coping
- Inheritance – ties class with its parent and may have limits on the performance of superclass
- Interfaces – allow to create faster implementation in the future

## Don’t warp API to achieve good performance

Problems with performance may go away in future releases, bad API – never

## Measure performance before and after each attempted optimization

Effect can be measurable or even negative

## Java doesn’t have a strong *performance model*

There is a gap between

*what the programmer writes*     and     *what the CPU executes*

## Item 56: Adhere to generally accepted naming conventions

## Packages

- Components of package names should consist of lowercase alphabetic characters and, rarely digits.
- Components should be short, generally eight or fewer characters.
- Meaningful abbreviations are encouraged
  (For example, *util* rather than *utilities*)

## Classes and interfaces

- Abbreviations are to be avoided, except for acronyms and certain common abbreviations like *max* and *min*.
- Acronyms can be either uppercase or have only their first letter capitalized
  (The second is better. *HTTPURL* or *HttpUrl*)

## Method and field names

- The same as classes’ names but the firstLetterShouldBeLowercase.
- If constant – UPPERCASE_WORDS_SEPARETED_WITH_UNDERSCORE

## Local variable

- The same as members’ names but abbreviations are available

## Type parameter names

- **T** for an arbitrary type
- **E** for the element type of a collection
- **K** and **V** for the key and value types of a map
- **X** for an exception

- A sequence of arbitrary types can be **T**, **U**, **V** or **T1**, **T2**, **T3**.