{{lastupdated_at}} by {{lastupdated_by}}

Method-call overheads

The method-call overhead is the time spent in passing arguments to a method and retrieving the return value. Normally, the method-call overhead is only relevant for methods that perform very little computations, so that a substantial fraction of the time is spent by setting up the method call.

To illustrate the method-call overhead, check the small attached program attachment:arguments.cpp / attachment:arguments.hpp. The program can be compiled using for example

```
$ g++ -o arguments arguments.cpp
```

The program tests passing 5 or 15 arguments in 3 different ways:

- Passing 5 or 15 int arguments by value
- Passing 5 or 15 int arguments by reference
- Passing 5 or 15 int arguments collected in a single structure by reference (which comes back to passing a single argument)

The table below gives the execution time for 100000000 (one hundred million) method calls for 5 or 15 arguments on a 2.66 GHz Intel Core i7 Mac OS X 10.6.8 system in seconds.

	5	**15**
By value	0.86	1.73
By reference	0.70	1.64
Using a structure	0.30	0.34

Obviously, passing a single structure as argument produces the smallest method-call overhead. Passing by reference is a little faster than passing by value, even for int values.

The general rules derived from this experiment are:

1. **Arguments should be passed by reference.**
2. **The number of arguments should be as small as possible. When necessary, group the arguments in a single structure.**

Note that 2. is even valid if the arguments change before each call. This has been verified by repeating the above experiment with setting of the arguments each time before the method is called. Here the resulting execution times in seconds:

	5	**15**
By value	1.02	2.15
By reference	0.83	2.19
Using a structure	0.46	0.98

Files

arguments.cpp	11 KB	03/26/2013	Knödlseder Jürgen
arguments.hpp	5.71 KB	03/26/2013	Knödlseder Jürgen