CPU PROFILING
FIND THE BOTTLENECK
WHAT? WHEN? HOW?
WHAT IS PROFILING?

- A form of dynamic analysis that measures some aspect of the program execution, typically:
  - Memory usage
  - Resource usage
  - Frequency and duration of function calls
When to Profile?

“We should forget about small efficiencies, say about 97% of the time: premature optimization is the root of all evil. Yet we should not pass up our opportunities in that critical 3%.”

- Donald Knuth
Tools

- "Optimization by guesswork" – bad!
- Hardcoded time measurement and logging
- Profilers
PROFILERS

- Sampling (statistical)
- Instrumenting
  - Source instrumenting
  - Code instrumenting
- (Event based)
- (Hypervisor)
TOOLS
AQTIME

- Delphi, C++ Builder, .NET (incl. Silverlight), Java ...
- Integration with RAD Studio and Visual Studio – D2006 and newer
- 32- and 64- bit
- Comes with XE7 and previous (limited version)
  - Additional downloads for registered users
- 539 €
Performance profiler
Allocation (memory) profiler
Coverage profiler
Static analysis profiler
Load library tracer profiler
More ...
www.prodelphi.de
Delphi 5 – XE7
32- and 64- bit
Very precise profiling
Free version (20 procedures)
Separate Ansi and Unicode version
Separate 32- and 64- bit version
50 – 90 €
SAMPLING PROFILER

- [http://delphitools.info/samplingprofiler](http://delphitools.info/samplingprofiler)
- Delphi 5 – XE4 (officially), works with XE7
- Measures time spent in OS DLLs
- Works at line level
- Real-time monitor
- Free
ASMPROFILER

- https://code.google.com/p/asmprofiler/
- Sampling profiler
- Instrumenting profiler
  - Add _uAsmProfDllLoader to program
- Usually more accurate results than Sampling Profiler
- Free
- Not limited to Delphi
DIY

- Home-brewed timing and logging
- GetTickCount
- Now
- timeGetTime
- QueryPerformanceCounter
- RDTSC
FIXING PERFORMANCE PROBLEMS
FIXING PERFORMANCE PROBLEMS

- Better algorithm 😊
  - Less memory allocations
  - Less string manipulations
  - Using different Windows controls

- Faster code 😞
  - Code optimization
  - Handcrafted assembler; using MMX/SSE

- Assembler tricks will not make up for bad design, however, they can make good design go faster.
- Distributed algorithms (GUI, messaging) are hard to profile
- Optimizing the inner code of an infinite loop doesn’t help
- If time is spent in kernel, reason may be hard to find
HANDS-ON!
QUESTIONS?