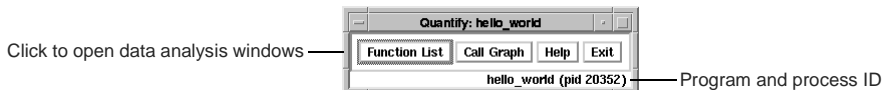# Rational® Quantify® Quick Reference

## Using Quantify

The CONTROL PANEL appears by default when running a Quantify'd program.
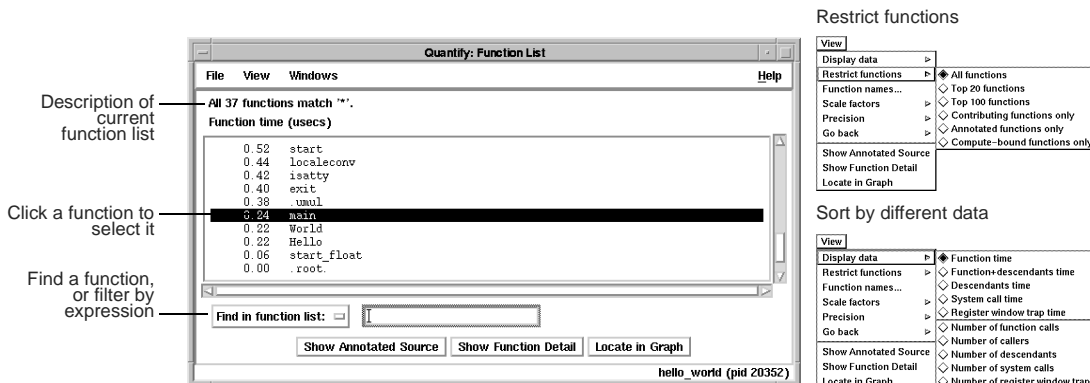You can also invoke `qv` on a saved `.qv` file:

```
%  qv a.out.23.0.qv
```

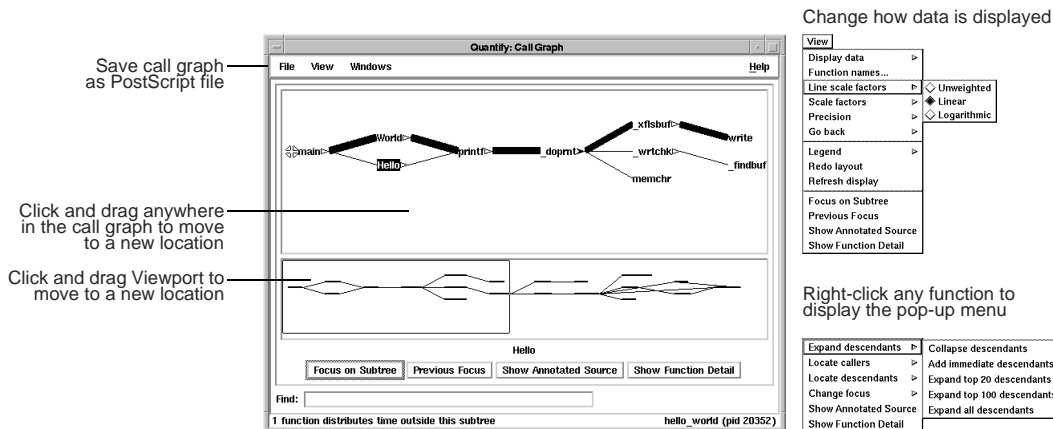## Control Panel: Open data analysis windows

Click to open data analysis windows —

**Quantify: hello_world**

| Function List | Call Graph | Help | Exit |

hello_world (pid 20352) — Program and process ID

## Function List window: Sort functions to find bottlenecks

Restrict functions

**Quantify: Function List**

File    View    Windows                                                 Help

Description of current function list — All 37 functions match '*'.

Function time (usecs)

```
0.52  start
0.44  localeconv
0.42  isatty
0.40  exit
0.38  .umul
0.24  main
0.22  World
0.22  Hello
0.06  start_float
0.00  .root.
```

Click a function to select it —

Find a function, or filter by expression —

Find in function list:   [          ]

| Show Annotated Source | Show Function Detail | Locate in Graph |

hello_world (pid 20352)

**View**
- Display data ▷
- Restrict functions ▷
- Function names... ▷
- Scale factors ▷
- Precision ▷
- Go back ▷
- Show Annotated Source
- Show Function Detail
- Locate in Graph

- ◆ All functions
- ◇ Top 20 functions
- ◇ Top 100 functions
- ◇ Contributing functions only
- ◇ Annotated functions only
- ◇ Compute-bound functions only

Sort by different data

**View**
- Display data ▷
- Restrict functions ▷
- Function names... ▷
- Scale factors ▷
- Precision ▷
- Go back ▷
- Show Annotated Source
- Show Function Detail
- Locate in Graph

- ◆ Function time
- ◇ Function+descendants time
- ◇ Descendants time
- ◇ System call time
- ◇ Register window trap time
- ◇ Number of function calls
- ◇ Number of callers
- ◇ Number of descendants
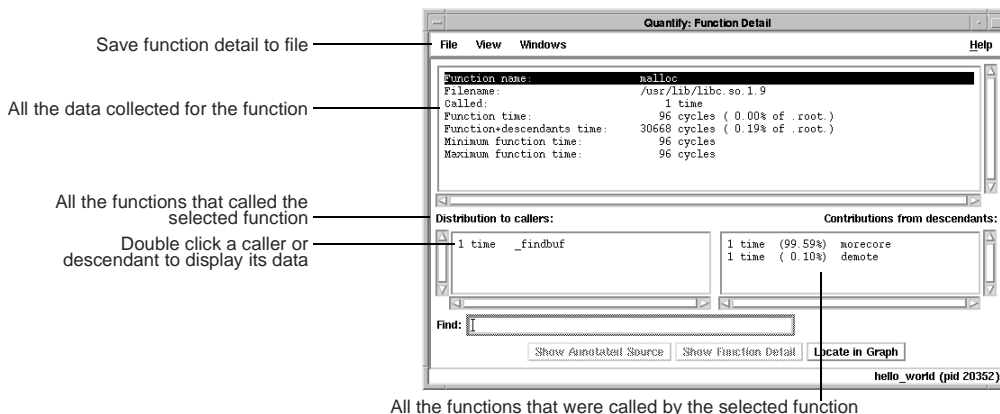- ◇ Number of system calls
- ◇ Number of register window traps

## Call Graph window: Understand your application's calling structure

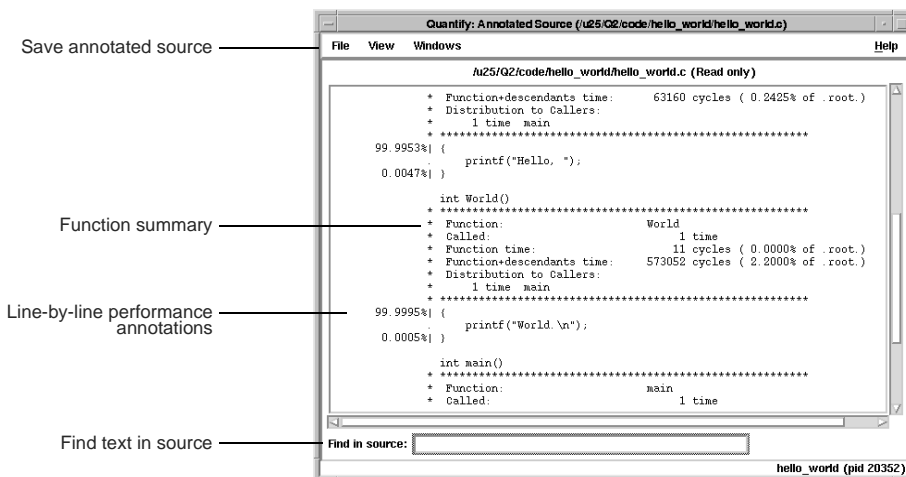By default, Quantify expands the top 20 descendents of `.root`.

Save call graph as PostScript file —

**Quantify: Call Graph**

File    View    Windows                                                 Help

Click and drag anywhere in the call graph to move to a new location —

Click and drag Viewport to move to a new location —

Hello

| Focus on Subtree | Previous Focus | Show Annotated Source | Show Function Detail |

Find: [          ]

1 function distributes time outside this subtree          hello_world (pid 20352)

Change how data is displayed

**View**
- Display data ▷
- Function names...
- Line scale factors ▷
- Scale factors ▷
- Precision ▷
- Go back ▷
- Legend ▷
- Redo layout
- Refresh display
- Focus on Subtree
- Previous Focus
- Show Annotated Source
- Show Function Detail

- ◇ Unweighted
- ◆ Linear
- ◇ Logarithmic

Right-click any function to display the pop-up menu

- Expand descendants ▷
- Locate callers ▷
- Locate descendants ▷
- Change focus ▷
- Show Annotated Source
- Show Function Detail

- Collapse descendants
- Add immediate descendants
- Expand top 20 descendants
- Expand top 100 descendants
- Expand all descendants

# Rational® Quantify® Quick Reference

## Function Detail window: Examine how a function's calling time is distributed

Save function detail to file

All the data collected for the function

All the functions that called the selected function

Double click a caller or descendant to display its data

```
Quantify: Function Detail

File    View    Windows                                              Help

Function name:               malloc
Filename:                    /usr/lib/libc.so.1.9
Called:                              1 time
Function time:                      96 cycles ( 0.00% of .root.)
Function+descendants time:       30568 cycles ( 0.19% of .root.)
Minimum function time:              96 cycles
Maximum function time:              96 cycles

Distribution to callers:                    Contributions from descendants:

1 time  _findbuf                            1 time (99.59%)  morecore
                                            1 time ( 0.10%)  demote

Find: [                                                              ]

        Show Annotated Source   Show Function Detail   Locate in Graph

                                                hello_world (pid 20352)
```

All the functions that were called by the selected function

## Annotated Source window: View line-by-line performance data

The Annotated Source window is available if you compile your program using the `-g` debugging option.

Save annotated source

Function summary

Line-by-line performance annotations

Find text in source

```
Quantify: Annotated Source (/u25/Q2/code/hello_world/hello_world.c)

File    View    Windows                                              Help

                  /u25/Q2/code/hello_world/hello_world.c (Read only)

    * Function+descendants time:      63160 cycles ( 0.2425% of .root.)
    * Distribution to Callers:
    *    1 time  main
    * *******************************************************
99.9953%| {
     .        printf("Hello, ");
  0.0047%| }

         int World()
    * *******************************************************
    * Function:                    World
    * Called:                         1 time
    * Function time:                 11 cycles ( 0.0000% of .root.)
    * Function+descendants time:  573052 cycles ( 2.2000% of .root.)
    * Distribution to Callers:
    *    1 time  main
    * *******************************************************
99.9995%| {
     .        printf("World.\n");
  0.0005%| }

         int main()
    * *******************************************************
    * Function:                    main
    * Called:                         1 time

Find in source: [                                                   ]

                                                hello_world (pid 20352)
```

## Conversion characters for filenames

Use these conversion characters when specifying filenames for options.

| Character | Converts to | Character | Converts to |
|---|---|---|---|
| %d | Current date (yymmdd) | %t | Current time (hh:mm:ss) |
| %p | Process id (pid) | %v | Program name |
| %n | Sequence number, starting at 0. Increments each time the dataset is saved. | %V | Full pathname of the program with "/" replaced by "_" |
| %T | Thread identifier | | |

# Rational® Quantify® Quick Reference

## API functions

Include `<quantifyhome>/quantify.h` in your code and link with `<quantifyhome>/quantify_stubs.a`

| Commonly used functions | Description |
|---|---|
| `quantify_help (void)` | Prints description of Quantify API functions |
| `quantify_is_running  (void)` | Returns `true` if the executable is Quantify'd |
| `quantify_print_recording_state (void)` | Prints the recording state of the process |
| `quantify_save_data (void)` | Saves data from the start of the program or since last call to `quantify_clear_data` |
| `quantify_save_data_to_file (char * filename)` | Saves data to a file you specify |
| `quantify_add_annotation (char * annotation)` | Adds the specified string to the next saved dataset |
| `quantify_clear_data (void)` | Clears the performance data recorded to this point |
| `quantify_<action>†_recording_data (void)` | Starts and stops recording of all data |
| `quantify_<action>†_recording_dynamic_library_data (void)` | Starts and stops recording dynamic library data |
| `quantify_<action>†_recording_register_window_traps (void)` | Starts and stops recording register-window trap data |
| `quantify_<action>†_recording_system_call` `(char *system_call_string)` | Starts and stops recording specific system-call data |
| `quantify_<action>†_recording_system_calls (void)` | Starts and stops recording of all system-call data |

**†** `<action>` is one of: `start`, `stop`, `is`. For example: `quantify_stop_recording_system_call`


## Build-time options

Specify build-time options on the link line to build Quantify'd programs:
```
% quantify -cache-dir=$HOME/cache -always-use-cache-dir cc ...
```

| Commonly used build-time options | Default |
|---|---|
| `-always-use-cache-dir`<br>   Specifies whether Quantify'd files are written to the global cache directory | no |
| `-cache-dir`<br>   Specifies the global cache directory | `<quantifyhome>/cache` |
| `-collection-granularity`<br>   Specifies the level of collection granularity | `line` |
| `-collector`<br>   Specifies the collect program to handle static constructors in C++ code | not set |
| `-ignore-runtime-environment`<br>   Prevents the run-time Quantify environment from overriding option values<br>   used in building the program | no |
| `-linker`<br>   Specifies an alternative linker to use instead of the system linker | system-dependent |
| `-use-machine`<br>   Specifies the build-time analysis of instruction times according to a particular machine | system-dependent |

# Rational® Quantify® Quick Reference

## qv run-time options

To run `qv`, specify the option and the saved `.qv` file:  `% qv -write-summary-file a.out.23.qv`

| qv options | Default |
|---|---|
| `-add-annotation`<br>    Specifies a string to add to the binary file | not set |
| `-print-annotations`<br>    Writes the annotations to `stdout` | no |
| `-windows`<br>    Controls whether Quantify runs with the graphical interface | yes |
| `-write-export-file`<br>    Writes the recorded data in the dataset to a file in export format | not set |
| `-write-summary-file`<br>    Writes the program summary for the dataset to a file | not set |

## Run-time options

Specify run-time options using the environment variable `QUANTIFYOPTIONS`:
```
% setenv QUANTIFYOPTIONS "-windows=no"; a.out
```

| Commonly used run-time options | Default |
|---|---|
| `-avoid-recording-system-calls`<br>    Avoids recording specified system calls | system-dependent |
| `-measure-timed-calls`<br>    Specifies measurement for timing system calls | elapsed-time |
| `-record-child-process-data`<br>    Records data for child processes created by `fork` and `vfork` | no |
| `-record-system-calls`<br>    Records system calls | yes |
| `-report-excluded-time`<br>    Reports time that was excluded from the dataset | 0.5 |
| `-run-at-exit`<br>    Specifies a shell script to run when the program exits | not set |
| `-run-at-save`<br>    Specifies a shell script to run each time the program saves counts | not set |
| `-save-data-on-signals`<br>    Saves data on fatal signals | yes |
| `-save-thread-data`<br>    Saves composite or per-stack thread data | composite |
| `-write-export-file`<br>    Writes the dataset to an export file as ASCII text | none |
| `-write-summary-file`<br>    Writes the program summary for the dataset to a file | /dev/tty |
| `-windows`<br>    Specifies whether Quantify runs with the graphical interface | yes |