QuickCheck Ready Reference

Properties

?FORALL(X,Generator,Property)—bind X to the generated value for use in the property.
?IMPLIES(Precondition,Property)—skip test cases where the precondition is false.
?WHENFAIL(Action,Property)—perform the Action when printing a counterexample to the property.
?TRAPEXIT(Property)—test Property in a process which traps exits. Do not use ?FORALL inside ?TRAPEXIT.
fails(Property)—succeeds if Property fails.
numtests(N,Property)—test the Property N times.
collect(X,Property)—test the property, and afterwards display the values of X.

Generators

?LET(X,Generator,…X…)—bind X to the generated value for use in the third argument.
?SUCHTHAT(X,Generator,Condition)—generate an X satisfying the condition.
?SIZED(Size,Generator)—bind Size to QuickCheck's size parameter for use in the generator.
resize(Size,Generator)—set the size parameter within the generator.
?LAZY(Generator)—build the generator only if it is chosen for use.
oneof([Gen1, Gen2, …])—choose one of the generators from a list.
frequency([{Weight1,Gen1},{Weight2,Gen2}…])—weighted choice of a generator from a list.
[X || Condition]++List—add X to List if the Condition is true.
elements(List)—choose an element from the list.
int()—a small integer.
nat()—a small non-negative integer.
real()—a real number.
bool()—true or false.
char()—a printable character.
choose(M,N)—an integer in the range M to N.
list(Generator)—a list of generated values.
default(Default,Generator)—specify a default value for the generator, which shrinking will select if possible.
weighted_default({Weight1,Default}, {Weight2,Generator})—same as default, with a weighted choice.

Fault Injection

fault(FaultyGenerator,CorrectGenerator)—define a fault that can be injected.
less_faulty(N,Generator)—inject each fault in the generator 1/Nth as often.
fault_rate(M,N,Generator)—inject faults in the generator in M out of N cases.

Shrinking

?SHRINK(Generator,[Gen1,Gen2…])—use Generator, then during shrinking try Gen1, Gen2…
noshrink(Generator)—use Generator, but disable shrinking of the result.

Symbolic Tests

Symbolic function calls: {call,Module,Function,Arguments}.
eval(SymbolicExpression)—evaluate symbolic function calls.

State Machines

Symbolic commands: {set,{var,V}, {call,Module,Function,Arguments}}.
Callbacks:

- initial_state()
- command(State)—generate a suitable command
- next_state(State,Result,Call)—state after the call
- precondition(State,Call)—is the call valid in a test?
- postcondition(State,Call,Result)—is the result correct?

commands(Module)—generate a list of commands from the callback module.
commands(Module,Init)—commands generated from an initial state.
run_commands(Module,Commands)—run the commands, returns {History,FinalState,Result}.
more_commands(N,Generator)—increase the number of commands generated by a factor N.

Interaction

eqc:quickcheck(Property)—run (by default) 100 tests of the property.
eqc:recheck()—repeat the last failing test.
eqc:watch_shrinking()—repeat the last failing test, and display cases tested during shrinking.
eqc:counterexample()—the last failing test case.
eqc:check(Property,CounterExample)—test the property in the given case.
eqc:module(Module)—test all the properties in the named module.
eqc_gen:sample(Generator)—display samples from the generator.
eqc_gen:sampleshrink(Generator)—display a sample and possible shrinkings.

QuviQ