VERIFICATION ACADEMY

UVM Basics

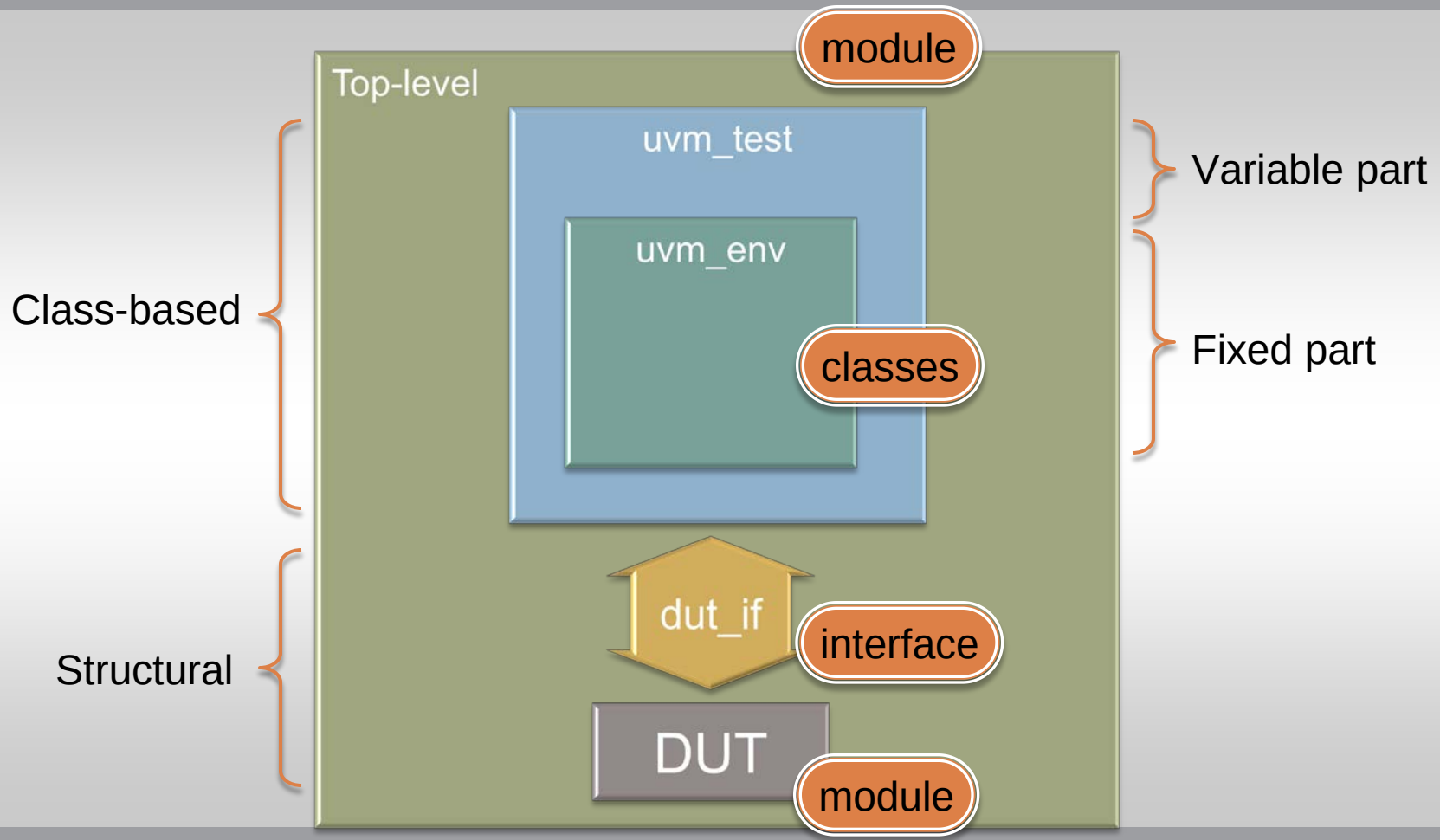
UVM "Hello World"

John Aynsley
CTO, Doulos

academy@mentor.com
www.verificationacademy.com

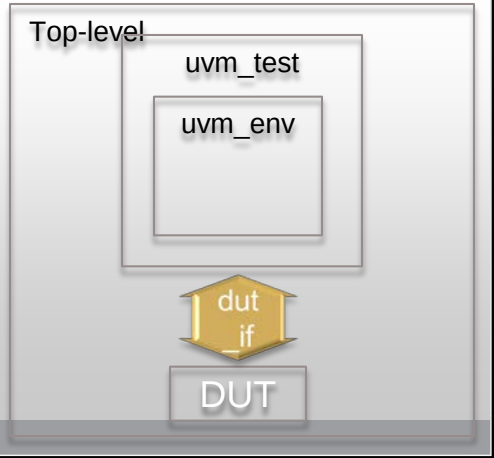
Mentor Graphics

DUT and Verification Environment



Interface

```
interface dut_if();

   ...

endinterface: dut_if
```

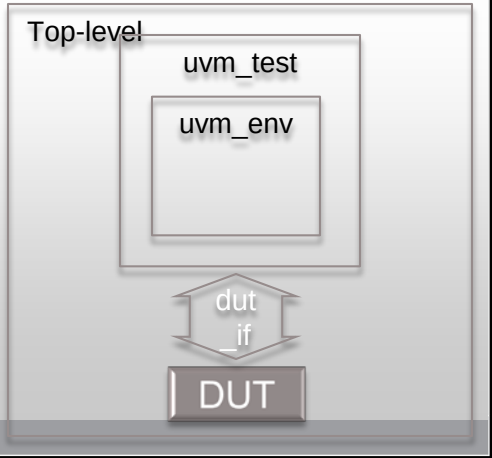


DUT

```
module dut(dut_if _if);
   ...
endmodule: dut
```

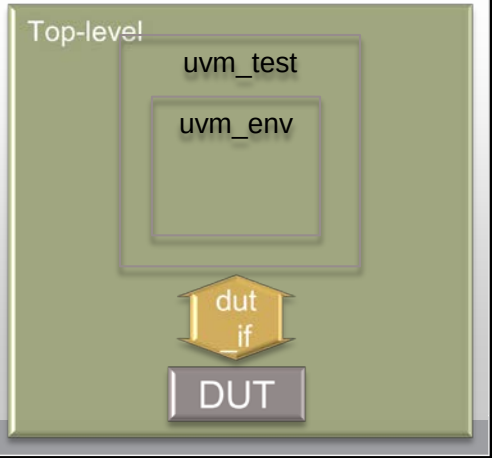
Interface port

or DUT wrapper



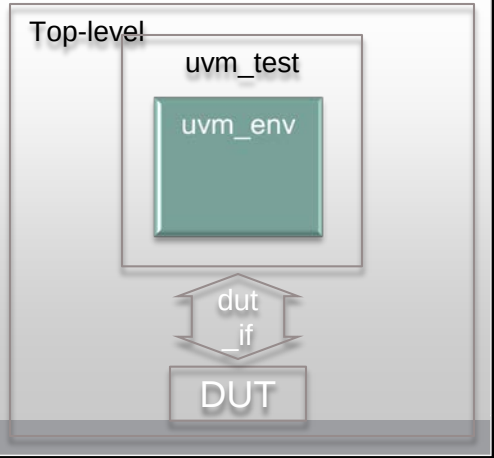
DUT Instantiation

```
module top;
  ...

  dut_if dut_if1 ();

  dut    dut1 ( ._if(dut_if1) );

  ...
endmodule: top
```



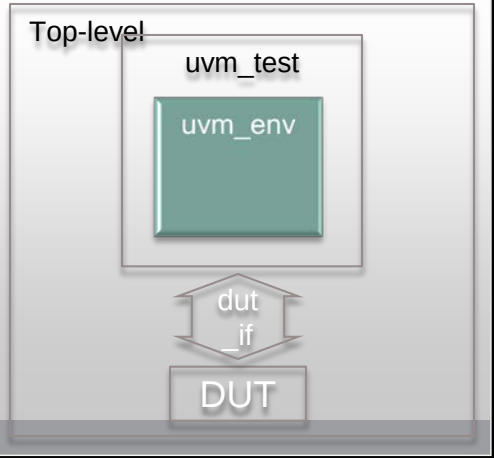
Env

```
class my_env extends uvm_env;



endclass: my_env
```



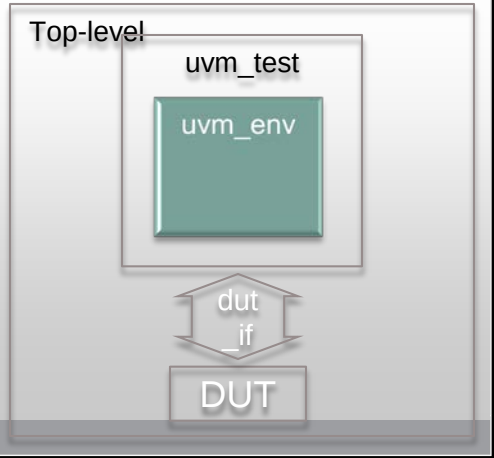
Env

```
class my_env extends uvm_env;
  `uvm_component_utils(my_env)




endclass: my_env
```



Env

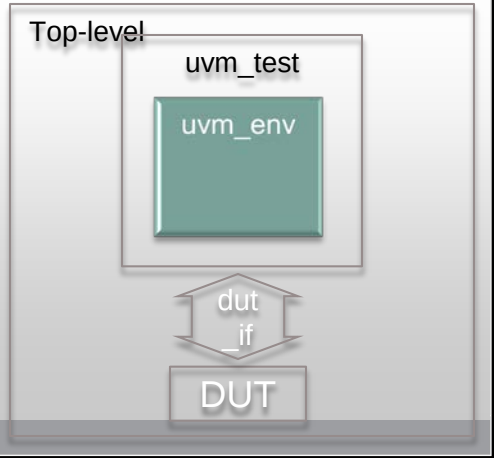
```
class my_env extends uvm_env;
  `uvm_component_utils(my_env)

  function new(string name, uvm_component parent);
    super.new(name, parent);
  endfunction: new




endclass: my_env
```



Env

```
class my_env extends uvm_env;
  `uvm_component_utils(my_env)

  function new(string name, uvm_component parent);
    super.new(name, parent);
  endfunction: new

  function void build_phase(uvm_phase phase);
    super.build_phase(phase);
  endfunction: build_phase

endclass: my_env
```

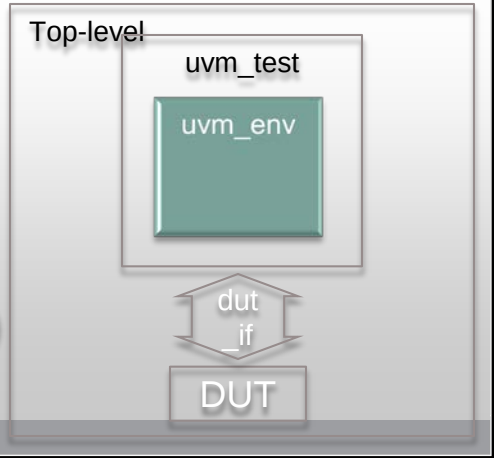
"Elaboration" in Verilog/VHDL



Env

```
class my_env extends uvm_env;
  `uvm_component_utils(my_env)

  function new(string name, uvm_component parent);
    super.new(name, parent);
  endfunction: new

  function void build_phase(uvm_phase phase);
    super.build_phase(phase);
  endfunction: build_phase

  task run_phase(uvm_phase phase);

  endtask: run_phase
endclass: my_env
```

"Simulation" in Verilog/VHDL



End-of-Test Mechanism

```
task run_phase(uvm_phase phase);
  phase.raise_objection(this);
  #10;
  phase.drop_objection(this);
endtask: run_phase
```

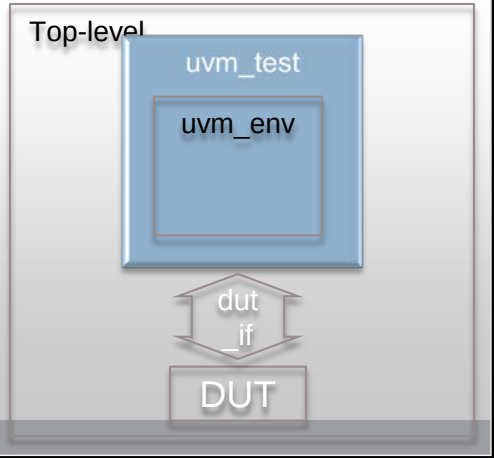
1st objection raised when time = 0

Consume some time

Test ends when all objections dropped

Test

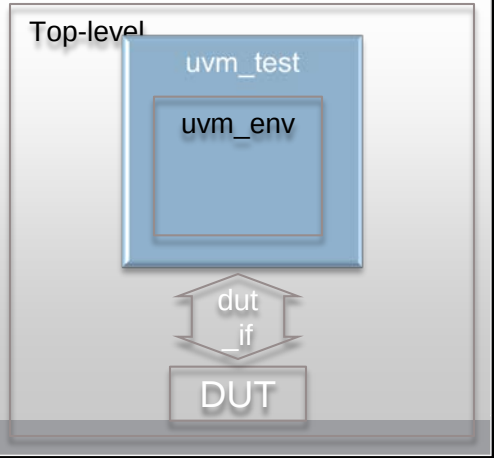
```
class my_test extends uvm_test;
  `uvm_component_utils(my_test)




endclass: my_test
```



Test

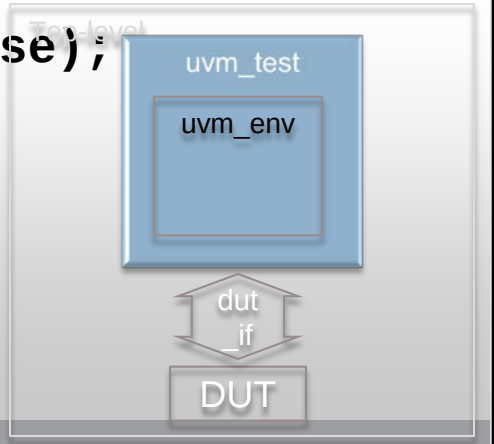
```
class my_test extends uvm_test;
  `uvm_component_utils(my_test)

  my_env my_env_h;

endclass: my_test
```

_h = handle



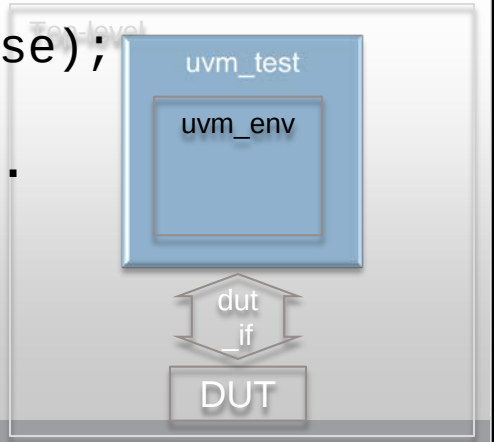
Test

```
class my_test extends uvm_test;
  `uvm_component_utils(my_test)

  my_env my_env_h;

  function new(string name, uvm_component parent);
    super.new(name, parent);
  endfunction: new

  function void build_phase(uvm_phase phase);
    super.build_phase(phase);
    ...
  endfunction: build_phase

endclass: my_test
```



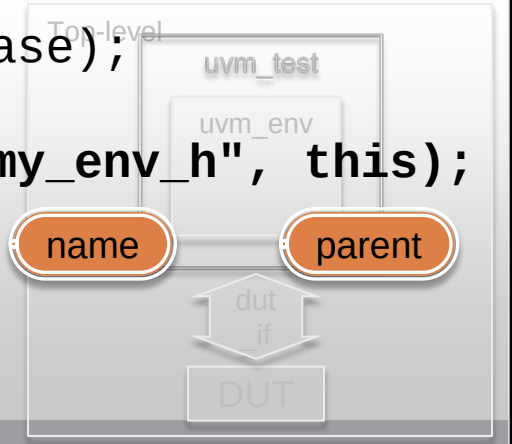
Test

```
class my_test extends uvm_test;
  `uvm_component_utils(my_test)

  my_env my_env_h;

  function new(string name, uvm_component parent);
    super.new(name, parent);
  endfunction: new

  function void build_phase(uvm_phase phase);
    super.build_phase(phase);
    my_env_h = my_env::type_id::create(...
  endfunction: build_phase

endclass: my_test
```



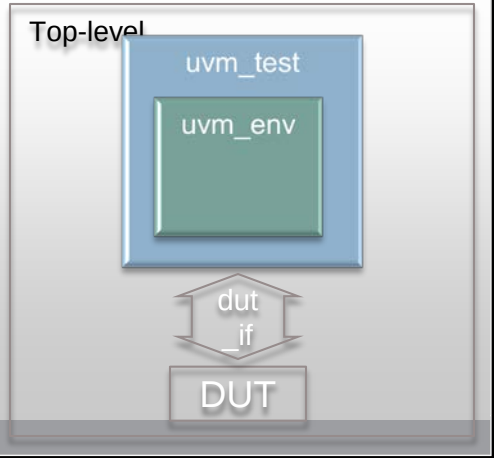
Test

```
class my_test extends uvm_test;
  `uvm_component_utils(my_test)

  my_env my_env_h;

  function new(string name, uvm_component parent);
    super.new(name, parent);
  endfunction: new

  function void build_phase(uvm_phase phase);
    super.build_phase(phase);
    my_env_h = my_env::type_id::create("my_env_h", this);
  endfunction: build_phase

endclass: my_test
```



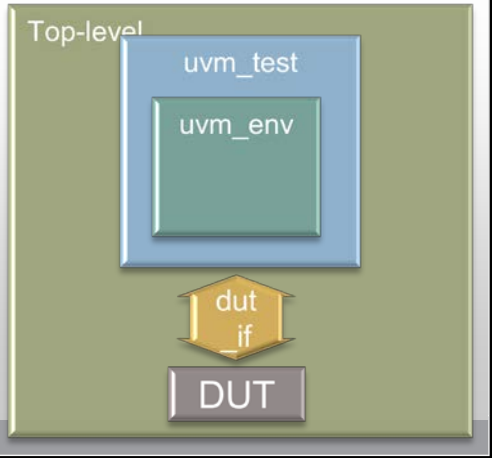
Package

```
`include "uvm_macros.svh"

package my_pkg;

  import uvm_pkg::*;

  class my_env extends uvm_env;
    ...
  endclass: my_env

  class my_test extends uvm_test;
    ...
  endclass: my_test

endpackage: my_pkg
```



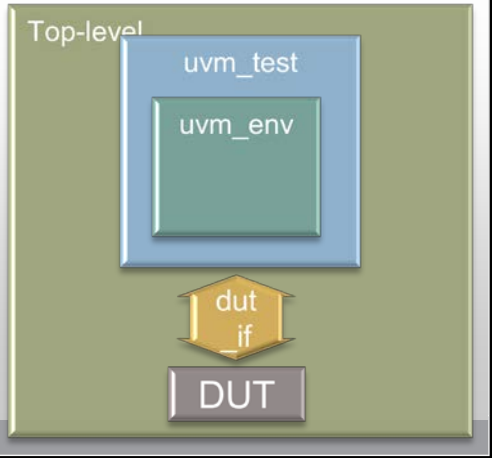
Test Instantiation

```
module top;

  import uvm_pkg::*;
  import my_pkg::*;

  dut_if dut_if1 ();

  dut    dut1 ( ._if(dut_if1) );




endmodule: top
```



Test Instantiation

```
module top;

  import uvm_pkg::*;
  import my_pkg::*;

  dut_if dut_if1 ();

  dut    dut1 ( ._if(dut_if1) );

  initial
  begin
    run_test("my_test");
  end

endmodule: top
```



Running the Simulation

```
qverilog file.sv
# Loading sv_std.std
# Loading work.uvm_pkg
# Loading work.my_pkg
# Loading work.top
# Loading work.dut_if
# Loading work.dut
# Loading ./work/_dpi/qv_dpi.so
# run -all
# ----------------------------------------------------------------
# UVM-1.0p1
# (C) 2007-2011 Mentor Graphics Corporation
# (C) 2007-2011 Cadence Design Systems, Inc.
# (C) 2006-2011 Synopsys, Inc.
# ----------------------------------------------------------------
```

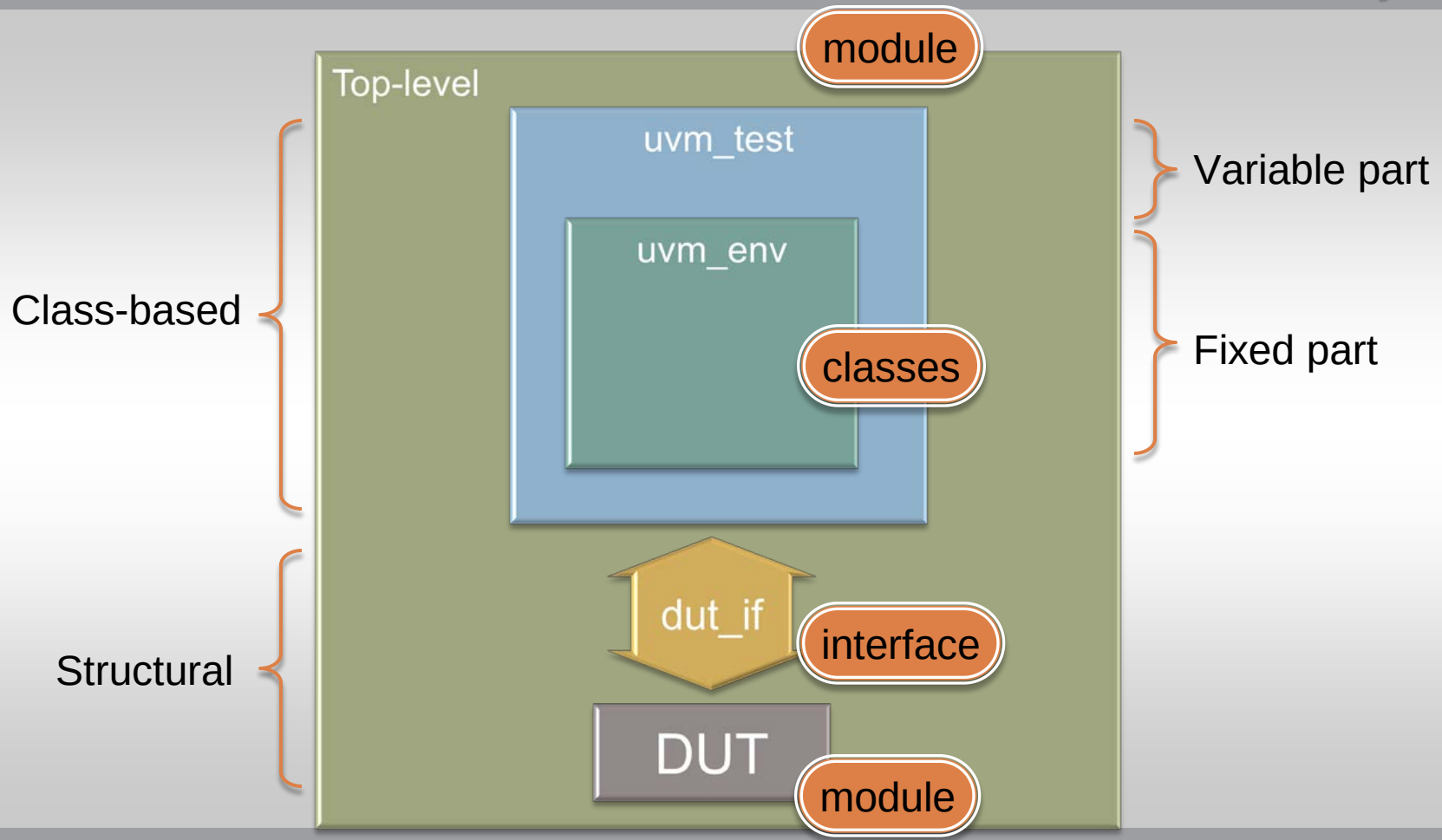
Running the Simulation

```
# UVM_INFO @ 0: reporter [RNTST] Running test my_test...
# UVM_INFO /home/sv0/ja/UVM/uvm-
1.0p1/src/base/uvm_objection.svh(1116) @ 10: reporter [TEST_DONE]
'run' phase is ready to proceed to the 'extract' phase
#
```

Running the Simulation

```
# --- UVM Report Summary ---
#
# ** Report counts by severity
# UVM_INFO :    2
# UVM_WARNING :    0
# UVM_ERROR :    0
# UVM_FATAL :    0
# ** Report counts by id
# [RNTST]     1
# [TEST_DONE]     1
# ** Note: $finish    : /home/sv0/ja/UVM/uvm-1.0p1/src/base/uvm_root.svh(392)
#    Time: 10 ns  Iteration: 44  Instance: /top
```

Summary


module
Top-level
uvm_test
uvm_env
classes
dut_if
interface
DUT
module
Class-based
Structural
Variable part
Fixed part

VERIFICATION ACADEMY

UVM Basics

UVM "Hello World"

John Aynsley
CTO, Doulos

academy@mentor.com
www.verificationacademy.com

Mentor Graphics