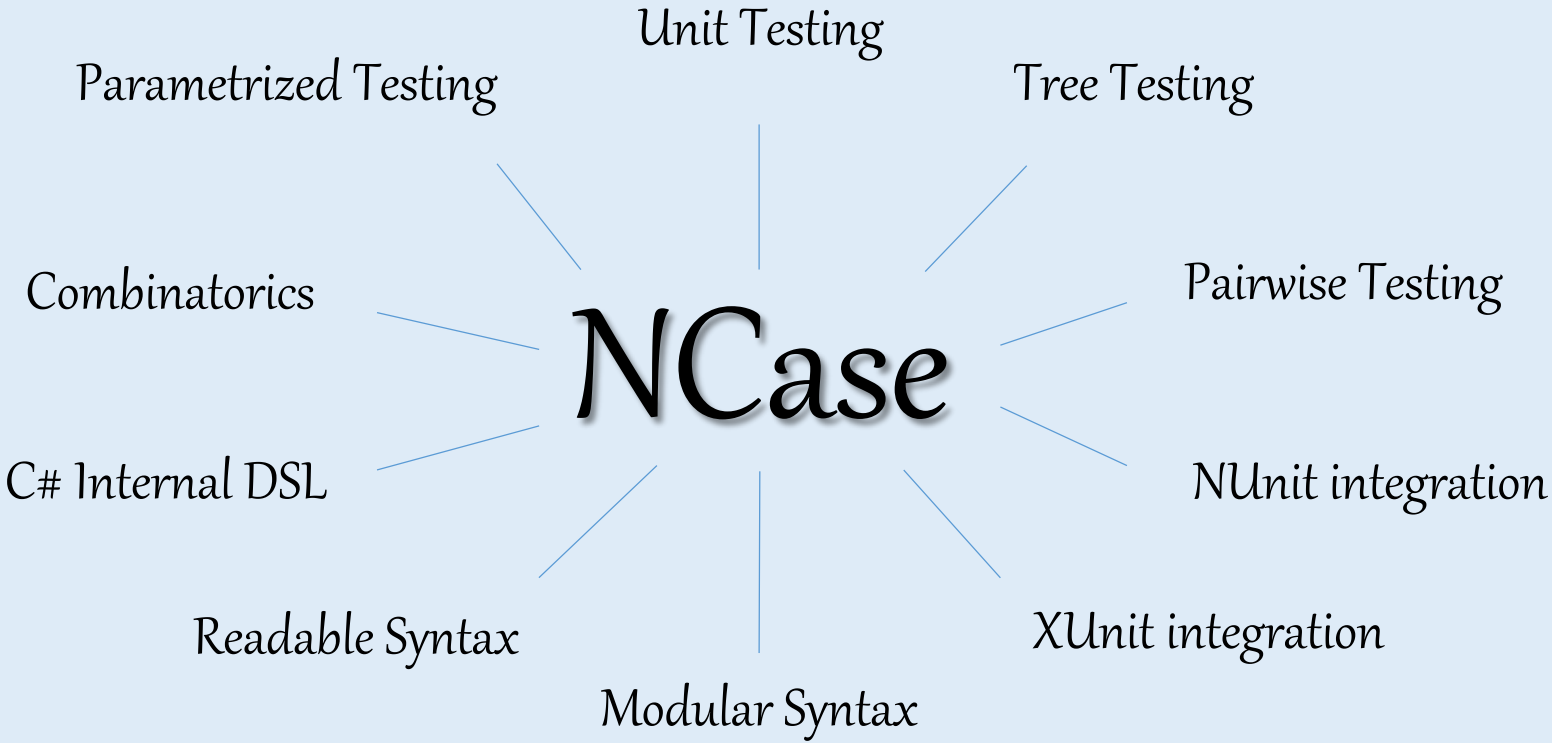
NCase

Test Case Generator


Unit Testing
Parametrized Testing
Tree Testing
Combinatorics
NCase
Pairwise Testing
C# Internal DSL
NUnit integration
Readable Syntax
XUnit integration
Modular Syntax

How do you currently write a Test?

Usual Test

ARRANGE

```
// ARRANGE
var mock = new Mock<ITodo>();
mock.SetupAllProperties();
ITodo todo = mock.Object;

todo.Title = "Remember to forget";

todo.DueDate = now;

todo.IsDone = false;
```

ACT

```
// ACT
var tm = new TodoManager();
bool ok = tm.AddTodo(todo);
```

ASSERT

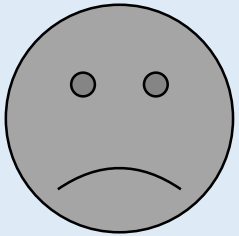
```
// ASSERT
Assert.IsTrue(ok);
Assert.AreEqual(1, tm.Todos.Count());
```

How do you currently write multiple Tests?

1st solution: Copy & Paste & Change

```
// ARRANGE
var mock = new Mock<ITodo>();
mock.SetupAllProperties();
ITodo todo = mock.Object;

todo.Title = "Remember to forget";

todo.DueDate = now;

todo.IsDone = false;

// ACT
var tm = new TodoManager();
bool ok = tm.AddTodo(todo);

// ASSERT
Assert.IsTrue(ok);
Assert.AreEqual(1, tm.Todos.Count());
```

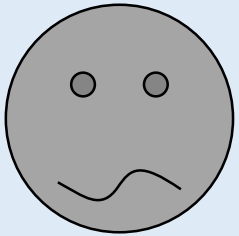
- Difficult to maintain
- No overview
- Low “Test Case Coverage”



2nd solution: parametrized test framework

```
[Test, Combinatorial]
public void ParametrizedTest(
    [Values("Remember to forget",
            "forget to remember",
            "and so on...", "all...",
            "and everything ...")] string title,
    [Values("yesterday",
            "now",
            "invalidLocalTime",
            "ambiguousLocalTime")] string dueDate,
    [Values(false, true)] bool isDone
    )
{
    // ARRANGE
    var mock = new Mock<ITodo>();
    mock.SetupAllProperties();
    ITodo todo = mock.Object;

    todo.Title = title;

    // Conversion due to attribute restrictions
    todo.DueDate = ConvertDueDate(dueDate);

    todo.IsDone = isDone;

    // ACT
    var tm = new TodoManager();
    bool ok = tm.AddTodo(todo);

    // ASSERT
    Assert.IsTrue(ok);
    Assert.AreEqual(1, tm.Todos.Count());
}
```

- Requires refactoring
- Various syntaxes with bad trade-off
  Scalability vs. Readability



Solution with

NCase

from Usual Test to NCase

Usual Test

NCase

ARRANGE

```
// ARRANGE
var mock = new Mock<ITodo>();
mock.SetupAllProperties();
ITodo todo = mock.Object;

todo.Title = "Remember to forget";

todo.DueDate = now;

todo.IsDone = false;
```

```
// ARRANGE
var builder = NCase.NewBuilder();
var todo = builder.NewContributor<ITodo>("todo");
var todoSet = builder.NewCombinationSet("todoSet");

using (todoSet.Define())
{
    todo.Title = "Remember to forget";

    todo.DueDate = now;

    todo.IsDone = false;
}

todoSet.Cases().Replay().ActAndAssert(ea =>
{
```

ACT

```
// ACT
var tm = new TodoManager();
bool ok = tm.AddTodo(todo);
```

```
    // ACT
    var tm = new TodoManager();
    bool ok = tm.AddTodo(todo);
```

ASSERT

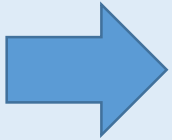
```
// ASSERT
Assert.IsTrue(ok);
Assert.AreEqual(1, tm.Todos.Count());
```

```
    // ASSERT
    Assert.IsTrue(ok);
    Assert.AreEqual(1, tm.Todos.Count());
});
```

from One Test to Many Tests

NCase One Test

```
// ARRANGE
var builder = NCase.NewBuilder();
var todo = builder.NewContributor<ITodo>("todo");
var todoSet = builder.NewCombinationSet("todoSet");

using (todoSet.Define())
{
    todo.Title = "Remember to forget";

    todo.DueDate = now;

    todo.IsDone = false;
}

todoSet.Cases().Replay().ActAndAssert(ea =>
{
    // ACT
    var tm = new TodoManager();
    bool ok = tm.AddTodo(todo);

    // ASSERT
    Assert.IsTrue(ok);
    Assert.AreEqual(1, tm.Todos.Count());
});
```



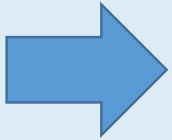
NCase Many Tests

```
// ARRANGE
var builder = NCase.NewBuilder();
var todo = builder.NewContributor<ITodo>("todo");
var todoSet = builder.NewCombinationSet("todoSet");

using (todoSet.Define())
{
    todo.Title = "Remember to forget";
    todo.Title = "forget to remember";
    todo.Title = "and so on...";
    todo.Title = "all...";
    todo.Title = "and everything ...";

    todo.DueDate = yesterday;
    todo.DueDate = now;
    todo.DueDate = invalidLocalTime;
    todo.DueDate = ambiguousLocalTime;

    todo.IsDone = false;
    todo.IsDone = true;
}

todoSet.Cases().Replay().ActAndAssert(ea =>
{
    // ACT
    var tm = new TodoManager();
    bool ok = tm.AddTodo(todo);
```

from One Test to Many Tests

NCase generates all possible combinations of

- 1 Title
- 1 Due Date
- 1 IsDone value



Many tests

```
// ARRANGE
var builder = NCase.NewBuilder();
var todo = builder.NewContributor<ITodo>("todo");
var todoSet = builder.NewCombinationSet("todoSet");

using (todoSet.Define())
{
    todo.Title = "Remember to forget";
    todo.Title = "forget to remember";
    todo.Title = "and so on...";
    todo.Title = "all...";
    todo.Title = "and everything ...";

    todo.DueDate = yesterday;
    todo.DueDate = now;
    todo.DueDate = invalidLocalTime;
    todo.DueDate = ambiguousLocalTime;

    todo.IsDone = false;
    todo.IsDone = true;
}

todoSet.Cases().Replay().ActAndAssert(ea =>
{
    // ACT
    var tm = new TodoManager();
    bool ok = tm.AddTodo(todo);
```

from One Test to Many Tests

#	todo.Title	todo.DueDate	todo.IsDone
1	Remember to forget	10.11.2011 00:00:00	False
2	Remember to forget	10.11.2011 00:00:00	True
3	Remember to forget	11.11.2011 00:00:00	False
4	Remember to forget	11.11.2011 00:00:00	True
5	Remember to forget	12.11.2011 00:00:00	False
6	Remember to forget	12.11.2011 00:00:00	True
7	Remember to forget	12.11.2011 00:00:00	False
8	Remember to forget	12.11.2011 00:00:00	True
9	forget to remember	10.11.2011 00:00:00	False
10	forget to remember	10.11.2011 00:00:00	True
11	forget to remember	11.11.2011 00:00:00	False
12	forget to remember	11.11.2011 00:00:00	True
13	forget to remember	12.11.2011 00:00:00	False
14	forget to remember	12.11.2011 00:00:00	True
15	forget to remember	12.11.2011 00:00:00	False
16	forget to remember	12.11.2011 00:00:00	True
17	and so on...	10.11.2011 00:00:00	False
18	and so on...	10.11.2011 00:00:00	True
19	and so on...	11.11.2011 00:00:00	False
20	and so on...	11.11.2011 00:00:00	True
21	and so on...	12.11.2011 00:00:00	False
22	and so on...	12.11.2011 00:00:00	True
23	and so on...	12.11.2011 00:00:00	False
24	and so on...	12.11.2011 00:00:00	True
25	all...	10.11.2011 00:00:00	False
26	all...	10.11.2011 00:00:00	True
27	all...	11.11.2011 00:00:00	False
28	all...	11.11.2011 00:00:00	True
29	all...	12.11.2011 00:00:00	False
30	all...	12.11.2011 00:00:00	True
31	all...	12.11.2011 00:00:00	False
32	all...	12.11.2011 00:00:00	True
33	and everything ...	10.11.2011 00:00:00	False
34	and everything ...	10.11.2011 00:00:00	True
35	and everything ...	11.11.2011 00:00:00	False
36	and everything ...	11.11.2011 00:00:00	True
37	and everything ...	12.11.2011 00:00:00	False
38	and everything ...	12.11.2011 00:00:00	True
39	and everything ...	12.11.2011 00:00:00	False
40	and everything ...	12.11.2011 00:00:00	True

TOTAL: 40 TEST CASES

How does

NCase

Work?

How does NCase work?

1

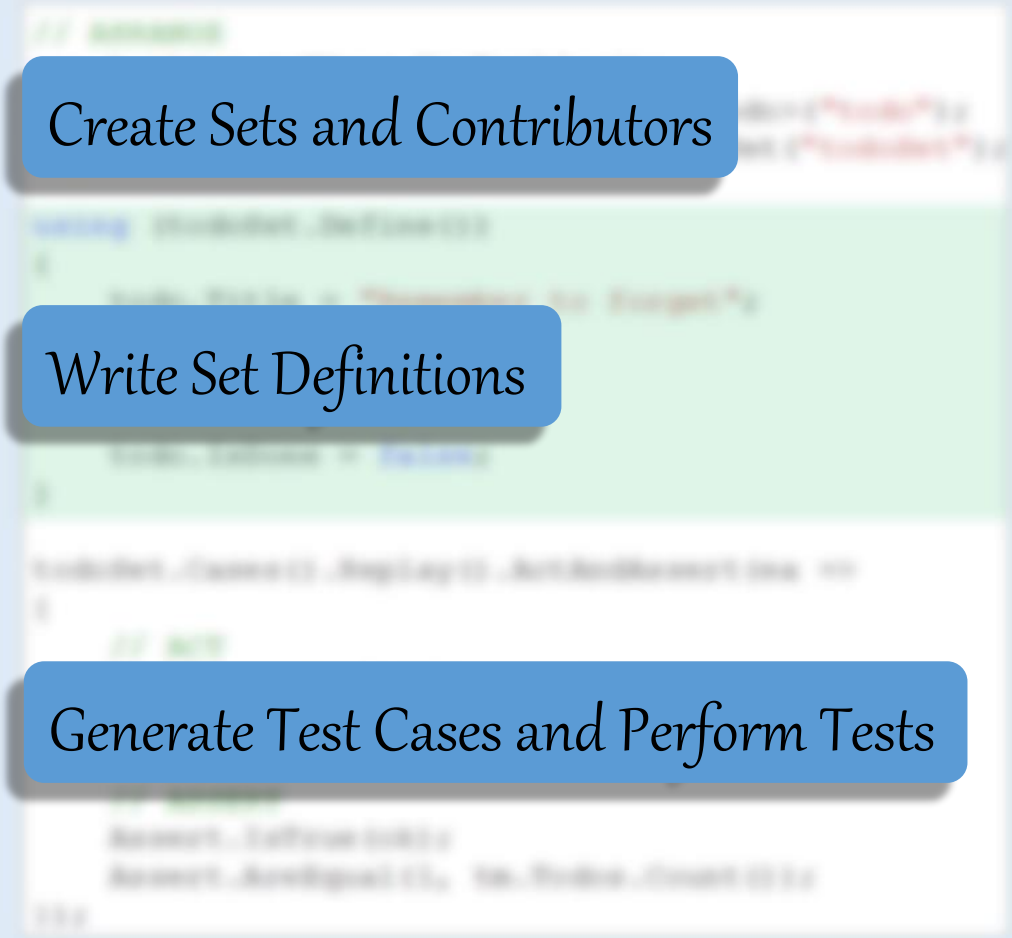
2

3

```
// ARRANGE
var builder = NCase.NewBuilder();
var todo = builder.NewContributor<ITodo>("todo");
var todoSet = builder.NewCombinationSet("todoSet");

using (todoSet.Define())
{
    todo.Title = "Remember to forget";

    todo.DueDate = now;

    todo.IsDone = false;
}

todoSet.Cases().Replay().ActAndAssert(ea =>
{
    // ACT
    var tm = new TodoManager();
    bool ok = tm.AddTodo(todo);

    // ASSERT
    Assert.IsTrue(ok);
    Assert.AreEqual(1, tm.Todos.Count());
});
```

How does NCase work?

1. Create Sets and Contributors
2. Write Set Definitions
3. Generate Test Cases and Perform Tests



How does NCase work?

1

Create Sets and Contributors

```
// ARRANGE
var builder = NCase.NewBuilder();
var todo = builder.NewContributor<ITodo>("todo");
var todoSet = builder.NewCombinationSet("todoSet");
```

2

- Create Builder
  builder owns all Sets and Contributors
- Create Contributors
  instances of any type, that you want to control

3

- Create Sets (here CombinationSet)
  defines a set of test cases

How does NCase work?

1

2

Write Set Definitions

3

```
using (todoSet.Define())
{
    todo.Title = "Remember to forget";

    todo.DueDate = now;

    todo.IsDone = false;
}
```

- Syntax specific to type of Set (DSL)
- Records all Contributor Calls

How does NCase work?

1

2

```
todoSet.Cases().Replay().ActAndAssert(ea =>
{
    // ACT
    var tm = new TodoManager();
    bool ok = tm.AddTodo(todo);

    // ASSERT
    Assert.IsTrue(ok);
    Assert.AreEqual(1, tm.Todos.Count());
});
```

- Generates all test cases
- Restores values in contributors
- Calls Act and Asserts

3

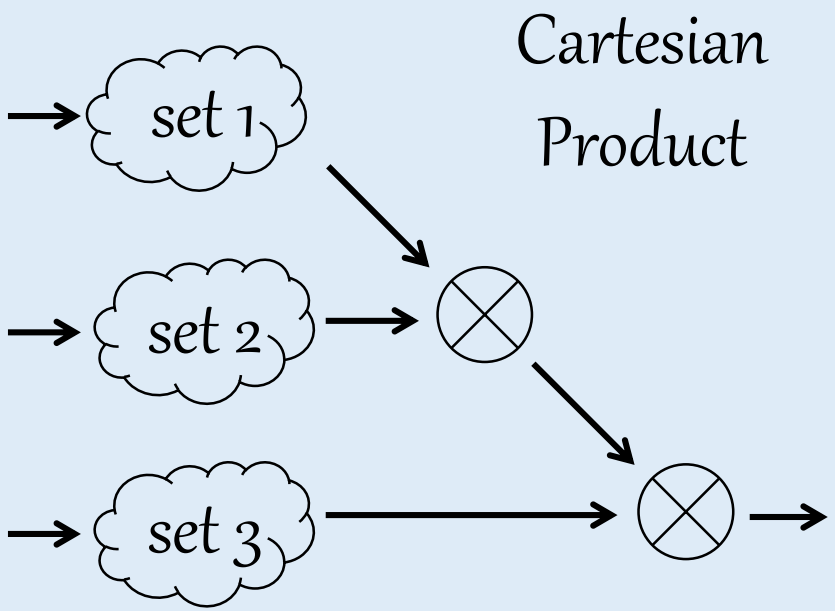
Generate Test Cases and Perform Tests

Combination Set

```
var todoSet = builder.NewCombinationSet("todoSet");
```

Cartesian Product

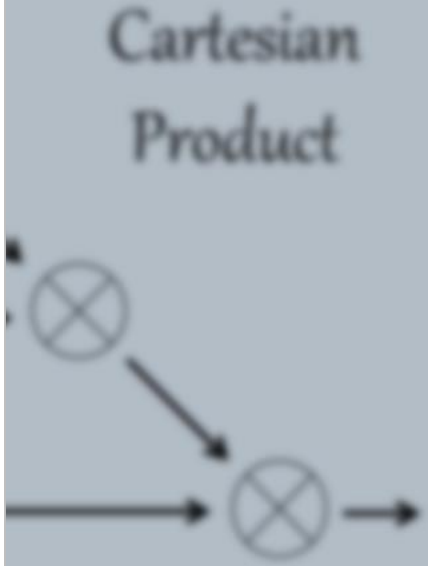
```
using (todoSet.Define())
{
    todo.Title = "Forget NCase";
    todo.Title = "Remember";
    todo.Title = "Love!";

    todo.DueDate = yesterday;
    todo.DueDate = now;
    todo.DueDate = tomorrow;

    todo.IsDone = false;
    todo.IsDone = true;
}
```



Cartesian Product

#	todo.Title	todo.DueDate	todo.IsDone
1	Forget NCase	10.11.2011 00:00:00	False
2	Forget NCase	10.11.2011 00:00:00	True
3	Forget NCase	11.11.2011 00:00:00	False
4	Forget NCase	11.11.2011 00:00:00	True
5	Forget NCase	12.11.2011 00:00:00	False
6	Forget NCase	12.11.2011 00:00:00	True
7	Remember	10.11.2011 00:00:00	False
8	Remember	10.11.2011 00:00:00	True
9	Remember	11.11.2011 00:00:00	False
10	Remember	11.11.2011 00:00:00	True
11	Remember	12.11.2011 00:00:00	False
12	Remember	12.11.2011 00:00:00	True
13	Love!	10.11.2011 00:00:00	False
14	Love!	10.11.2011 00:00:00	True
15	Love!	11.11.2011 00:00:00	False
16	Love!	11.11.2011 00:00:00	True
17	Love!	12.11.2011 00:00:00	False
18	Love!	12.11.2011 00:00:00	True

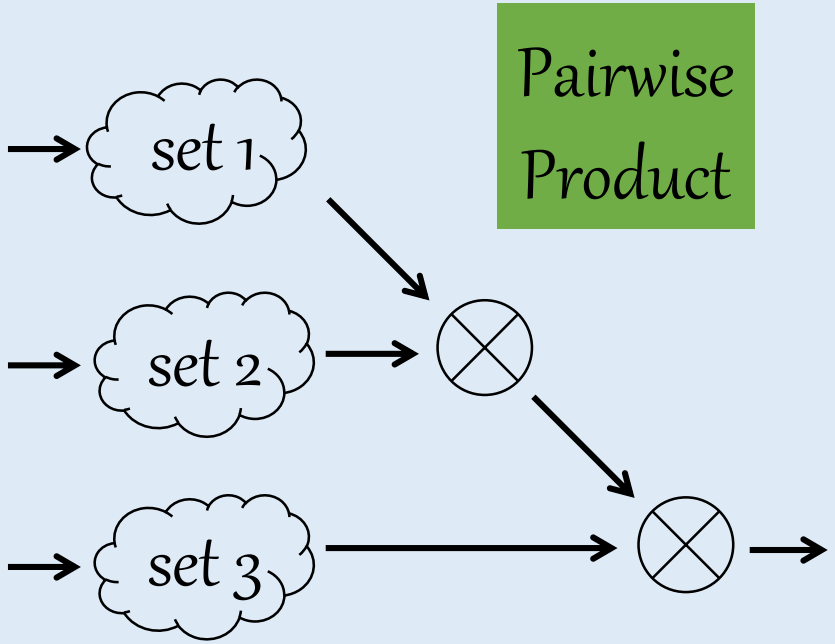
TOTAL: 18 TEST CASES



Pairwise Product

```
var todoSet = builder.NewCombinationSet("todoSet", onlyPairwise: true);
```

```
using (todoSet.Define())
{
    todo.Title = "Forget NCase";
    todo.Title = "Remember";
    todo.Title = "Love!";

    todo.DueDate = yesterday;
    todo.DueDate = now;
    todo.DueDate = tomorrow;

    todo.IsDone = false;
    todo.IsDone = true;
}
```



Pairwise Product

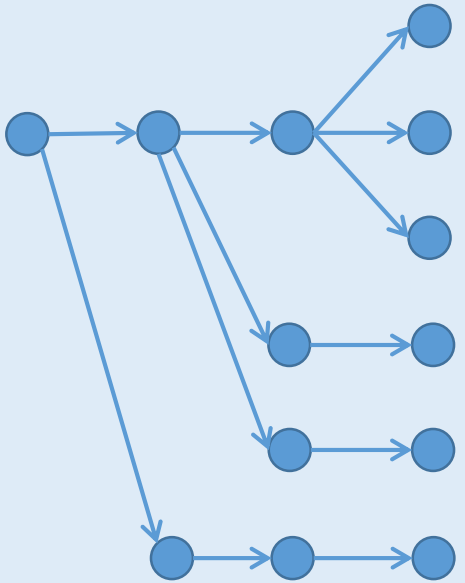
```
var todoSet = builder.NewCombinationSet("todoSet", onlyPairwise: true);
```

#	todo.Title	todo.DueDate	todo.IsDone
1	Forget NCase	10.11.2011 00:00:00	False
2	Forget NCase	11.11.2011 00:00:00	True
3	Forget NCase	12.11.2011 00:00:00	False
4	Remember	10.11.2011 00:00:00	False
5	Remember	11.11.2011 00:00:00	True
6	Remember	12.11.2011 00:00:00	True
7	Love!	10.11.2011 00:00:00	False
8	Love!	11.11.2011 00:00:00	True
9	Love!	12.11.2011 00:00:00	False
10	Forget NCase	10.11.2011 00:00:00	True
11	Forget NCase	11.11.2011 00:00:00	False

TOTAL: 11 TEST CASES

Combination Tree

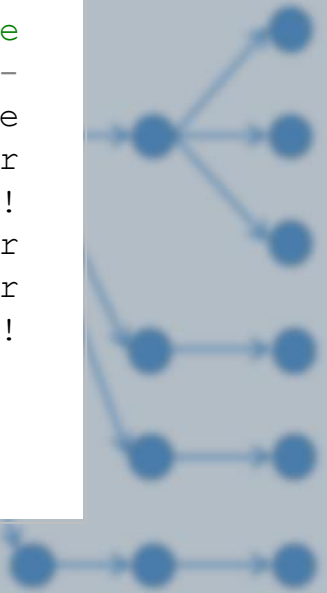
```
using (todoSet.Define())
{
    todo.IsDone = true;
        todo.DueDate = yesterday;
            todo.Title = "Forget NCase";
            todo.Title = "Remember";
            todo.Title = "Love!";
        todo.DueDate = now;
            todo.Title = "Remember";
        todo.DueDate = tomorrow;
            todo.Title = "Remember";
    todo.IsDone = false;
        todo.Title = "Love!";
            todo.DueDate = tomorrow;
}
```



Combination Tree

#	todo.IsDone	todo.DueDate	todo.Title
1	True	10.11.2011 00:00:00	Forget NCase
2	True	10.11.2011 00:00:00	Remember
3	True	10.11.2011 00:00:00	Love!
4	True	11.11.2011 00:00:00	Remember
5	True	12.11.2011 00:00:00	Remember
6	False	12.11.2011 00:00:00	Love!

TOTAL: 6 TEST CASES



Mix Contributors

You can mix contributors within definitions

```
var todo = builder.NewContributor<ITodo>("todo");

var user = builder.NewContributor<IUser>("user");

using (todoSet.Define())
{
    todo.Title = "Forget NCase";
    todo.Title = "Remember";

    todo.DueDate = now;
    todo.DueDate = tomorrow;

    user.Email = "some@email.com";

    user.IsActive = true;
    user.IsActive = false;
}
```

Mix Contributors

You can mix contributors within definitions

```
var todo

var user

using (t
{
    todo
    todo

    todo
    todo

    user.Email = "some@email.com";

    user.IsActive = true;
    user.IsActive = false;
}
```

#	todo.Title	todo.DueDate	user.Email	user.IsActive
1	Forget NCase	11.11.2011 00:00:00	some@email.com	True
2	Forget NCase	11.11.2011 00:00:00	some@email.com	False
3	Forget NCase	12.11.2011 00:00:00	some@email.com	True
4	Forget NCase	12.11.2011 00:00:00	some@email.com	False
5	Remember	11.11.2011 00:00:00	some@email.com	True
6	Remember	11.11.2011 00:00:00	some@email.com	False
7	Remember	12.11.2011 00:00:00	some@email.com	True
8	Remember	12.11.2011 00:00:00	some@email.com	False

TOTAL: 8 TEST CASES

Reference other Definitions

You can reference definitions within definitions

```
using (allSet.Define())
{
    todoSet.Ref();

    user.Email = "some@email.com";

    user.IsActive = true;
    user.IsActive = false;
}
```

Reference other Definitions

#	todoSet	user.Email	user.IsActive
1	X	some@email.com	True
2	X	some@email.com	False

TOTAL: 2 TEST CASES

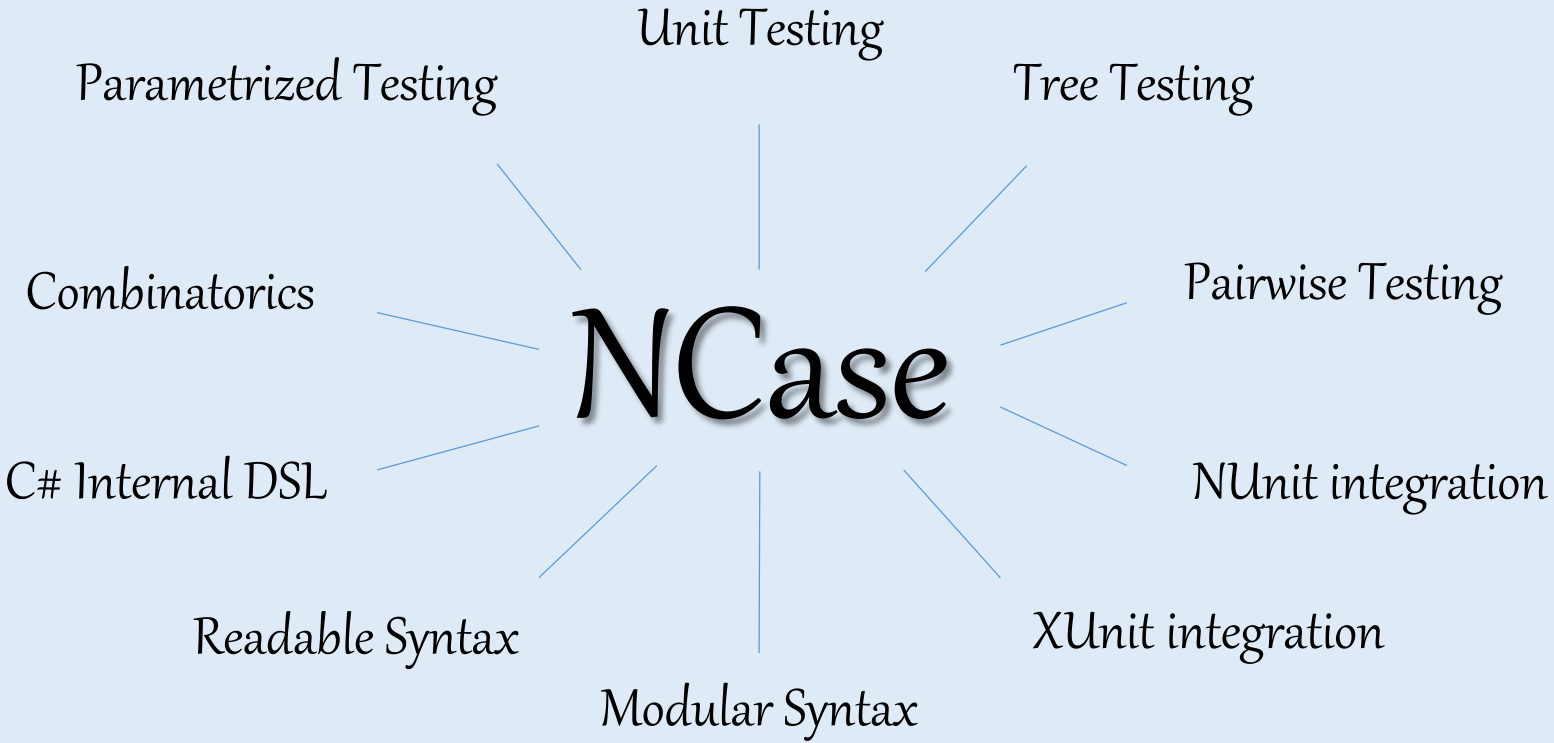
#	todo.Title	todo.DueDate	todo.IsDone	user.Email	user.IsActive
1	Forget NCase	10.11.2011 00:00:00	False	some@email.com	True
2	Forget NCase	10.11.2011 00:00:00	False	some@email.com	False
3	Forget NCase	10.11.2011 00:00:00	True	some@email.com	True
4	Forget NCase	10.11.2011 00:00:00	True	some@email.com	False
5	Forget NCase	11.11.2011 00:00:00	False	some@email.com	True
6	Forget NCase	11.11.2011 00:00:00	False	some@email.com	False
7	Forget NCase	11.11.2011 00:00:00	True	some@email.com	True
8	Forget NCase	11.11.2011 00:00:00	True	some@email.com	False
9	Forget NCase	12.11.2011 00:00:00	False	some@email.com	True
10	Forget NCase	12.11.2011 00:00:00	False	some@email.com	False
11	Forget NCase	12.11.2011 00:00:00	True	some@email.com	True
12	Forget NCase	12.11.2011 00:00:00	True	some@email.com	False
13	Remember	10.11.2011 00:00:00	False	some@email.com	True
14	Remember	10.11.2011 00:00:00	False	some@email.com	False
15	Remember	10.11.2011 00:00:00	True	some@email.com	True
16	Remember	10.11.2011 00:00:00	True	some@email.com	False
17	Remember	11.11.2011 00:00:00	False	some@email.com	True
18	Remember	11.11.2011 00:00:00	False	some@email.com	False
19	Remember	11.11.2011 00:00:00	True	some@email.com	True
20	Remember	11.11.2011 00:00:00	True	some@email.com	False
21	Remember	12.11.2011 00:00:00	False	some@email.com	True
22	Remember	12.11.2011 00:00:00	False	some@email.com	False
23	Remember	12.11.2011 00:00:00	True	some@email.com	True
24	Remember	12.11.2011 00:00:00	True	some@email.com	False
25	Love!	10.11.2011 00:00:00	False	some@email.com	True
26	Love!	10.11.2011 00:00:00	False	some@email.com	False
27	Love!	10.11.2011 00:00:00	True	some@email.com	True
28	Love!	10.11.2011 00:00:00	True	some@email.com	False
29	Love!	11.11.2011 00:00:00	False	some@email.com	True
30	Love!	11.11.2011 00:00:00	False	some@email.com	False
31	Love!	11.11.2011 00:00:00	True	some@email.com	True
32	Love!	11.11.2011 00:00:00	True	some@email.com	False
33	Love!	12.11.2011 00:00:00	False	some@email.com	True
34	Love!	12.11.2011 00:00:00	False	some@email.com	False
35	Love!	12.11.2011 00:00:00	True	some@email.com	True
36	Love!	12.11.2011 00:00:00	True	some@email.com	False

TOTAL: 36 TEST CASES

Coming Features

Coming Features

- Support recording of method calls
  (currently only property calls are recorded)
- Support class contributors
  (currently only interface contributors are supported)
- Moq like Setup(...) and Verify(...)
- Permutation Set
  permuting order of calls
- Dedicated unit test runner
  currently integrates with NUnit, XUnit. Dedicated runner would improve usability within IDE



http://github.com/jeromerg/NCase