

Pragmatic Unit Testing: Summary

The following checklists are extracted from the book *Pragmatic Unit Testing in C# with NUnit*, part of the Pragmatic Starter Kit series. More information is available at <http://www.pragmaticprogrammer.com/titles/utc2>, where you can also order PDF and paper copies of this book and our other titles.

General Principles:

- Test anything that might break.
- Test everything that does break.
- New code is guilty until proven innocent.
- Write at least as much test code as production code.
- Run local tests with each compile.
- Run all tests before check-in to the repository.

What to Test: Use Your “Right BICEP”

- Are the results **r**ight?
- Are all the **b**oundary conditions **C**ORRECT?
- Can you check **i**nverse relationships?
- Can you **c**ross-check results using other means?
- Can you force **e**rror conditions to happen?
- Are **p**erformance characteristics within bounds?

Questions to Ask:

- If the code ran correctly, how would I know?
- How am I going to test this?
- What *else* can go wrong?
- Could this same kind of problem happen anywhere else?

Good tests are “A TRIP”

- A**utomatic
- T**horough
- R**epeatable
- I**ndependent
- P**rofessional

CORRECT Boundary Conditions

- C**onformance: Does the value conform to an expected format?
- O**rdering: Is the set of values ordered or unordered as appropriate?
- R**ange: Is the value within reasonable minimum and maximum values?
- R**eference: Does the code reference anything external that isn't under direct control of the code itself?
- E**xistence: Does the value exist (for example, is non-null, non-zero, present in a set, and so on)?
- C**ardinality: Are there exactly enough values?
- T**ime (absolute and relative): Is everything happening in order? At the right time? In time?