文章
姚 鑫 · 五月 15, 2021 阅读大约需 4 分钟

第一章 单元测试概述

第一章 单元测试概述

本教程的第一部分概述了单元测试。完成本教程的这一部分后，将能够：
- 定义单元测试并区分单元测试和集成测试
- 列出单元测试的几个好处
- 描述InterSystems IRIS %UnitTest包和xUnit测试框架之间的相似性。
- 列出软件开发中测试优先方法经常声称的几个好处。

什么是单元测试?

单元测试是对单个代码模块的正确性的测试，例如，方法或类的测试。通常，开发人员在开发代码时为其代码创建单元测试。典型的单元测试是一种执行方法的方法，该方法测试并验证该方法是否为给定的一组输入生成了正确的输出。

单元测试不同于集成测试。集成测试验证了一组代码模块交互的正确性。单元测试仅单独验证代码模块的正确性。一组代码模块的集成测试可能会失败，即使每个模块都通过了单元测试。

为什么要进行单元测试？

单元测试提供了许多好处，包括：

- 提供代码模块是否正确的验证。这是单元测试的主要原因。
- 提供自动回归测试。更改代码模块后，应重新运行单元测试，以确保代码模块仍然正确。也就是说，应该使用单元测试来确保更改没有破坏代码模块。理想情况下，所有代码模块的单元测试都应该在更改任何一个模块之后运行。
- 提供文档。通常，代码模块的单元测试与代码模块一起交付。检查单元测试提供了大量有关代码模块如何工作的信息。

XUnit测试框架

单元测试框架是为开发和执行单元测试提供支持的类包。它们可以很容易地扩展以支持更具体或专门化类型的单元测试。

XUnit系列测试框架基于原始的Sunit框架(用于单元测试SmallTalk代码)，包括以下框架：

- JUnit-Java代码的单元测试框架。
- NUnit-C#、VB.NET和其他.NET语言代码的单元测试框架。
- CppUnit-C++代码的单元测试框架。
- PyUnit-Python代码的单元测试框架。

%UnitTest和xUnit框架的结构

%UnitTest包和xUnit框架共享相同的基本结构。熟悉任何Unit框架的开发人员都可以毫不费力地学习使用%UnitTest包。%UnitTest和xUnit框架都围绕以下基本测试结构组织：

- 测试装置-为一个测试或一组测试做准备和清理工作的代码。准备测试可能包括创建数据库连接，或使用测试数据初始化数据库。清理可能包括关闭数据库连接或恢复数据库状态。
- 测试用例-测试的最小单元。验证特定的一组输入是否会产生给定模块的特定输出。
- 测试套件-设计为一起执行的测试和测试套件的集合。
- Test Runner-用于执行测试并显示其结果的实用程序。

测试自动化

%UnitTest包和xUnit框架都支持测试自动化。当单元测试完成执行时，它会报告测试是通过还是失败。不需要解释测试结果。这是非常重要的。可以为每个代码更改执行大量单元测试。如果必须不断地阅读和解释结果，这个过程很快就会变得非常乏味和容易出错。

许多xUnit框架提供了汇总测试结果的图形用户界面(GUI)。%UnitTest会生成一个显示测试结果的网页。它以绿色显示有关通过的测试的信息，以红色显示有关失败的测试的信息。开发人员可以一目了然地判断是否有任何测试失败。

这是由%UnitTest单元测试生成的测试报告。用户可以通过单击页面上的超链接深入查看提供有关测试的更多详细信息的页面。

测试优先方法论

敏捷软件方法论，例如测试驱动开发(TDD)和极限编程，特别强调单元测试。事实上，这些方法使用单元测试来驱动开发过程。他们提倡“ 测试优先 ”的软件开发方法。在这种方法中，开发人员在编写代码模块的一行代码之前设计并编写代码模块的单元测试。然后，开发人员创建代码模块，目标是通过单元测试。

Test First方法的倡导者声称该方法具有以下好处：

- 它迫使开发人员在开发任何模块之前很久就决定代码模块的正确输入和输出。
- 它集中了开发人员在创建代码模块时的注意力。开发人员关注的是在创建模块时通过单元测试的具体目标。
- 它可以防止单元测试成为事后的想法。如果首先创建单元测试，则在项目结束之前不能忽略单元测试。
- 它确保了代码的高度测试覆盖率。

注意：测试优先开发的支持者通常主张在代码模块之前执行单元测试，而不仅仅是创建单元测试。当然，在这一点上测试应该会失败。他们甚至可能不会编译。

Red – Green – Refactor

XUnit和%UnitTest测试报告GUI报告以绿色表示通过测试，以红色表示未通过测试。下面是使用测试优先开发方法的开发节奏：
1. 红色 - 编写一个不起作用的小测试，也许一开始不会编译。
2. 绿色 - 让测试快速运行，在测试过程中犯下所有必要的错误。
3. 重构 - 消除仅在使测试正常工作时产生的所有重复。

> Kent Beck，《测试驱动的设计》

#SQL #Caché #InterSystems IRIS #InterSystems IRIS for Health

源 URL: https://cn.community.intersystems.com/post/%E7%AC%AC%E4%B8%80%E7%AB%A0-%E5%8D%95%E5%85%83%E6%B5%8B%E8%AF%95%E6%A6%82%E8%BF%B0