## windows下fitness python版本安装测试

 (2014-03-27 16:12:03)

[IMG_256](http://blog.sina.com.cn/s/javascript:;)转载▼

|  |  |
| --- | --- |
| 标签： [it](http://search.sina.com.cn/?c=blog&q=it&by=tag) | 分类： [工作](http://blog.sina.com.cn/s/articlelist_1141251645_3_1.html) |

### FitNesse介绍[¶](http://blog.sina.com.cn/s/blog_44061e3d0101j8ts.html" \l "FitNesse%E4%BB%8B%E7%BB%8D)

* FitNesse是一套软件开发协作工具。
  1. 伟大的软件需要协作和交流，FitNesse可以帮助大家加强软件开发过程中的协作。能够让客户、测试人员和开发人员了解软件要做成什么样，自动完成它实际是什么样，帮助建议软件最终是否达到了设计初衷。在项目早期的开发过程中，它是在早期对一些复杂问题解决（使他们正确）的很有价值的协作方式。
* FitNesse是一套软件测试工具。
  1. FitNesse是一个轻量级的、开源的框架，能够帮助开发团队方便的定义验收测试（Acceptance Tests），通过在web页面上简单的输出和预计输出的表格就可实现，并且可以运行这些测试以确定是否通过。
* FitNesse可以作为一个wiki,FitNesse wiki可以很方便的创建和编辑页面。
* FitNesse是一个Web服务器,不用过多的安装配置，很方便使用。它不需要配置，直接在机器上面运行就可以通过浏览器访问。

### windows下FitNesse安装[¶](http://blog.sina.com.cn/s/blog_44061e3d0101j8ts.html" \l "windows%E4%B8%8BFitNesse%E5%AE%89%E8%A3%85)

* 下载地址：http://www.fitnesse.org/FitNesseDownload，下载安装包fitnesse-standalone.jar
* 命令行安装
  1. cmd转到fitnesse.jar所在的目录，比如E:\me\fitnesse
  2. 运行命令：java -jar fitnesse-standalone.jar -p 8083，默认为80端口，如需其他端口，则-p加指定端口
  3. 通过http://localhost/ 来行访问fitnesse
* 也可以直接双击fitnesse-standalone.jar，会生成一个FitNesseRoot目录，第一次用时进入到FitNesseRoot，然后在浏览器就可以通过http://localhost/ 来行访问fitnesse了

### PyFIT下载安装[¶](http://blog.sina.com.cn/s/blog_44061e3d0101j8ts.html" \l "PyFIT%E4%B8%8B%E8%BD%BD%E5%AE%89%E8%A3%85)

* fitnesse支持多种语言版本，如果想用python编写测试脚本，则除了python环境外，还需要下载安装PyFIT，下载地址http://agile.unisonis.com/PyFitTutorial.html
* 下载PyFIT-0.8a2.zip，将其放在python安装目录中解压缩，进入目录 python setup.py install安装
* 具体使用步骤可查阅http://agile.unisonis.com/PyFitTutorial.html

### python示例[¶](http://blog.sina.com.cn/s/blog_44061e3d0101j8ts.html" \l "python%E7%A4%BA%E4%BE%8B)

* 编写python代码，比如在E:\me\fit目录下

#-\*- coding: utf-8 -\*-  
from fit.ColumnFixture import ColumnFixture  
  
class Hellofit(ColumnFixture):  
      \_typeDict = {  
        "descrip":"String",  
        "num1":"Int",  
        "num2":"Int",  
        "result":"Int"  
        }  
        
      def \_init\_(self):  
           ColumnFixture.\_\_init\_\_(self)  
           self.descrip=''  
           self.num1=''  
           self.num2=''  
             
      def result(self):  
           add=self.num1+self.num2  
           return add

* fitness中点击Edit，输入以下内容

!define COMMAND\_PATTERN {python "%m" %p}  
!define TEST\_RUNNER {C:\Python27\PyFIT-0.8a2\fit\FitServer.py}  
!path E:\me\fit  
  
!|Hellofit|  
|descrip|num1|num2|result?|  
|add|1|2|3

* save以后点击Test就可以看到测试结果了

通过以下总结的内容，想必会对想了解[测试](http://lib.csdn.net/base/softwaretest)的人以及刚进入测试的“猿”，有个总体的概括，希望能帮到大家！

**1.什么是接口测试**

接口测试是测试系统组件间接口的一种测试。接口测试主要用于检测外部系统与系统之间以及内部各个子系统之间的交互点。测试的重点是要检查数据的交换，传递和控制管理过程，以及系统间的相互逻辑依赖关系等。

**2.为什么做接口测试**

首先，节省测试成本，数据模型推算，底层的一个bug能够引发上层的8个左右bug，而且底层的bug很容易引起全网的宕机。相反接口测试能够提供系统复杂度上升情况下的低成本高效率的解决方案。  
其次接口测试不同于传统开发的单元测试，接口测试是站在用户的角度对系统接口进行全面高效持续的检测。  
最后接口测试是自动化并且持续集成的，这也是为什么接口测试能够低成本高收益的根源。  
总之接口测试是保证高复杂性系统质量的内在要求和低成本的经济利益的驱动作用下的最佳解决方案，接口测试是一个完整的体系，也包括功能测试、性能测试。

**3.接口测试的适用范围**

接口测试一般应用于多系统间交互开发，或者拥有多个子系统的应用系统开发的测试。接口测试适用于为其他系统提供服务的底层框架系统和中心服务系统，主要测试这些系统对外部提供的接口，验证其正确性和稳定性。接口测试同样适用于一个上层系统中的服务层接口，越往上层，其测试的难度越大。接口测试在淘宝的应用是一个自下而上的发展过程。  
接口测试实施在多系统多平台的构架下，有着极为高效的成本收益比。接口测试天生为高复杂性的平台带来高效的缺陷检测和质量监督能力。平台越复杂，系统越庞大，接口测试的效果越明显。

**4.在接口测试中如何应对需求的频繁变化**

在现在这个互联网软件时代，需求的频繁变动已经不是什么新鲜事。客户的需求变更、市场需求的变更，项目本身的调整，以及新需求的出现等等都会导致需求的变化。这种需求的变化常会出现在项目开发阶段，根据需求的变化开发人员会对项目进行调整，而作为在项目开发阶段就接入进行测试的接口测试人员同样也会被影响，这种影响有时是巨大的，影响着我们的工作效率，它会导致我们需要重复以前的部分测试工作，甚至会让我们以前所做的测试工作白费。而且越是大型的、复杂的项目，这种影响越大 ，暴露出的问题也越多。  
　　针对这段期间我在项目中的体验，将需求变化对接口测试的影响和出现的问题罗列下：  
　　1. 需求变化，接口测试人员不知道或过了很久才知道。由于某些原因，常常会导致新需求变动接口测试人员不知道，或是过了很久才知道。往往接口测试人员是通过用例回归发现用例跑不通，然后会进行错误排查，最后发现问题后和开发确认后才知道是需求变化。这样是很浪费时间，甚至会遗漏一些需要测试的新需求的功能点，导致测试不全，遗漏bug。  
　　2. 需求变化，对原有测试用例及其代码的影响.这个也是最让我头痛的、最直接的影响。需求变动有时会打乱了原有的测试规划，甚至包括对测试特性的划分原则，相应的测试结果分析验证、测试需求跟踪等都不到位。并且我们接口测试会对一个项目写上百个测试用例，为了尽可能的发现bug，测试用例里面有无数的验证点。往往一个很小的需求的改变会影响到很多的测试用例代码不通过，我们需要对很多测试用例进行调整，需要对测试数据以及测试代码进行修改，有时甚至需要修改我们的测试框架。这对我们接口测试人员来说是一个不少的挑战。  
　　3. 新需求变化测试时间短，开展详细的测试有难度。由于新需求的提出已在开发期间，其测试时期短，接口测试有时没有人力和时间投入对新增修改需求的测试分析和设计上，基本上很难像对待老需求一样，开展详细的测试分析设计。  
　　针对以上所写的这些，我说说我的拙见，如何减少需求变更对接口测试的影响：  
　　1. 良好的心态。从心态上，接口测试人员应该把需求变化当作是一种项目常态，平常心应对。但是，我们也要学会控制这种需求变化的趋势，不能任其发展。  
　　2. 及时沟通，最快知晓需求变更。和需求相关人员和开发人员做好即时沟通，第一时间知道需求的变更，及时做好测试策略更新。知道的越早对我们的影响越小，需要的测试成本也越低。  
　　3. 良好的团队合作。接口测试人员和开发人员的良好合作，分工明确，对新的改动及时通知对方，短时间内开展最有效的团队协作。接口测试人员要主动关注开发代码的修改，对测试用例和测试代码及时调整，做到小粒度的修改。  
　　4. 接口测试人员反应快，用例代码灵活性高。接口测试人员反应快，提前做好新需求的测试规划，包括测试设计和测试执行规划，并且在设计中要考虑新需求对老需求的影响；并且我们原测试用例和代码也要有一定的灵活，可以在一定程度上适应需求变化，将未来的新需求的影响尽量降到很小。这里就不详细说了，下次就具体的MC的项目说说如何增强测试用例代码的灵活性，减少新需求对测试代码的影响。  
　　5. 做到及时的需求跟踪。通过测试用例代码的不断回归，尽早的主动的发现需求的变更。  
我们接口测试人员要成熟、快速、有序、灵活、有责任心的应对需求的变化，把我们的接口测试工作做得更好。

**5.接口测试中测试与开发的配合**

作为一名测试人员，工作中接触最频繁的应该要数开发人员了。在整个测试过程中，开发人员是与测试人员是走的最近的，因为从最初测试的需求到测试中发现的缺陷的处理以及最终测试的总结，都需要和开发人员紧密合作。  
　　接口测试因其天生的代码亲密性，为了更好地提高产品质量，就要求测试人员更加地深入到开发的工作中去（从需求出发深入到代码、页面中去），甚至是与开发并行地工作。那么这就对测试人员和开发人员的合作与互动提出了更高的要求。  
　　1、测试与开发的互动应该贯穿项目始终，时刻保持和开发的联系  
　　也许有人会认为开发和测试在项目中相互独立会更加好，在此对这个问题不作讨论，只想说说从始至终与开发保持联系的好处。时刻保持联系，可以使双方对于项目的进展有一个明确的共同的理解，使项目的执行更加顺利。减少一些缺乏沟通而可能造成的工作内容的冲突，例如对于需求理解的不一致、需求变更等。  
　　2、测试需求不光来自于PRD和UC，还要倾听来自开发的需求，这往往是他们担心的内容  
　　诚然PRD和UC是测试需求的主要来源也是测试工作的依据，然而从PRD和UC出发的测试需求往往是功能性的，会遗漏不少细节，特别是在接口测试工作中，这些细节又往往体现在开发的工作中，或者某些具体的实现中。因此倾听开发的测试需求，同时提出测试对于开发的要求是十分必要的。开发的需求经常是体现了在开发中他们没有把握的地方，这些光靠分析PRD和UC是很难得到的。  
　　3、测试与开发应当相互了解对方的工作内容和方式，并交换意见  
　　让开发知道测试在做什么是怎么做的，当前测试的状况是什么样子的，测试也要了解开发的进度和工作内容。开发了解测试的方法和内容会有利于提高代码的可测性以及代码的品质。测试了解开发的工作方式和进度，就便于和开发进行合作加快缺陷的修复和验证。不仅要了解，在了解的基础上相互交换意见和看法，这往往能相互提高工作效率。  
　　4、职责明确，测试应全面负责测试的工作环境、配置、代码，开发不应当随意改动。  
　　讲了很多互动的地方，但是有些内容却不应是互动的。就接口测试来说，测试环境配置和测试代码应当全部由测试工程师来维护，因为测试工程师主导整个测试的过程并对测试的结果负责。可以请开发协助配置环境这是必要的，但是出现任何测试方面的问题，开发都不应该在没有和测试工程师沟通的情况下介入测试环境和测试代码的变更。因为这样往往会导致测试用例无法通过，测试环境被破坏，测试结果可信度下降。会给项目进度带来不必要的影响。  
　　相应的，由于接口测试的代码往往和工作代码在一个工程下，测试也不应该去改动开发的工作内容，这会带来十分严重的后果。  
　　5、测试应当高度关注测试持续集成的结果，第一时间分析问题，并初步定位后转交开发。  
　　在此我想强调下初步定位并转交开发的问题，可能有些同学会觉得缺陷被发现后直接转交开发就可以了，定位缺陷的事开发完成就可以了。我有一些不同的想法，能够定位缺陷意味着对于项目有着较为深入的了解，将有助于提高缺陷修复的效率。例如，同样是查询结果异常的问题，原因可能各有不同，如果初步将问题进行定位，必定能提高这些相似但实质却不同的缺陷的解决效率。

**6.如何简单设计接口测试用例**

接口测试是项目测试的一部分，它测试的主要对象是接口 ，是测试系统组件间接口的一种测试。接口测试主要用于检测外部系统与所测系统之间以及内部各系统之间的交互点。测试的重点是检查数据交互、传递、和控制管理过程以及系统间的相互依赖关系等。  
　　如何设计接口测试用例？首先，明确出发点，和所有的测试一样 ，接口测试出发点是你要证明所测的程序是错误的。以这个出发点为导向 ，你的设计行为就会尽量朝这个方向，更易发现问题  
　　其次，选择好测试对象。对于一个系统做接口测试选择好的测试对象是接口测试关键。一个系统有无数的接口 ，每个接口如果分别测试 ，那将是很痛苦的一件事情，而且任何一个内部接口的变动 ，都将导致我们用例的不可用。  
　　可将这些最外层的接口分为两类：一类是数据进入系统的接口；一类是数据流出系统的接口。进入系统的接口实际是我们用例的执行调用的接口。可通过变化参数对这些接口进行调用 ，模拟外部的使用；而流出的接口则是我们用例真正该验证的点。数据从哪里流出，流出时的状态如何 ，此时系统又是什么状态都是我们所应该验证的。  
　　然后，确认完整的测试对象的功能：确认外部接口提供给使用这些接口的外部用户什么样的功能，外部用户真正需要什么样的功能。此两个功能一定要准确详细，用例的设计要严格按照测试对象功能设计才是正确的用例。  
　　最后当出发点、对象、功能都确定了，就可以真正设计用例了。下面详细介绍下如何去设计一个结构好、可读性高、渗透性强的接口测试用例。  
　　接口测试用例设计和测试用例设计一样，用例设计的内容应该包括：主要测试功能点、测试环境、测试数据、执行操作以及预期结果。  
　　1）接口测试环境分为两种：一种是程序内部的环境；一种是程序的所调用外部接口的环境。  
　　2）接口测试测试数据分为接口参数数据和用例执行所需系统数据。数据的设计、准备测试用例的数据上需要花费更多的心思。要通过好的测试数据使用例查找问题。接口参数数据需对每个参数根据测试接口的实际的功能进行分析，在符合业务逻辑的情况下进行逻辑组合排列 ，不要遗漏了某些边界值和错误点的数据。每个用例执行所需系统数据和接口参数数据尽可能的采用不一样的数据 ，使用例更容易发现问题。  
　　3）测试功能点，如果一个接口功能复杂时推荐对接口用例进行结构划分 ，这样子用例具有更好的可读性和维护性。接口划分原则为以接口提供的功能点的不同进行合适粒度的划分。同一功能点的用例又可根据测试环境的不同、数据的不同进行用例的填充。  
    4）接口测试用例执行操作非常简单，就是所测接口的调用。

　　5）预期结果验证，这也是接口用例设计的很关键的一步 ，应该细而不冗余。每个用例均需验证 ，避免一个用例中重复做相同的验证 ，提高测试用例的效率。