

Keyword Driven Testing with GUIDancer

GUI tests, whether manual or automatic, consist of a few actions which are repeated again and again. Texts are entered into text fields, buttons are clicked, entries from combo boxes are selected etc. It makes sense that a tester shouldn't have to start from scratch to create, record or program these steps each time he begins automating a test. These essential building blocks or modules should already be available for him to reuse and reference to create larger modules. The libraries of test cases in GUIDancer allow just this. Tests can be created simply by dragging and dropping test cases from the library.

What's in a name?

The modules in a keyword-driven test are called keywords because of their name. It should be easy to identify what the keyword does, and what data it already contains, just by reading the name. Naming keywords according to conventions makes the test more readable and makes tests easier and quicker to create - it is easy to find the keyword you need based just on the name.

Hierarchical tests

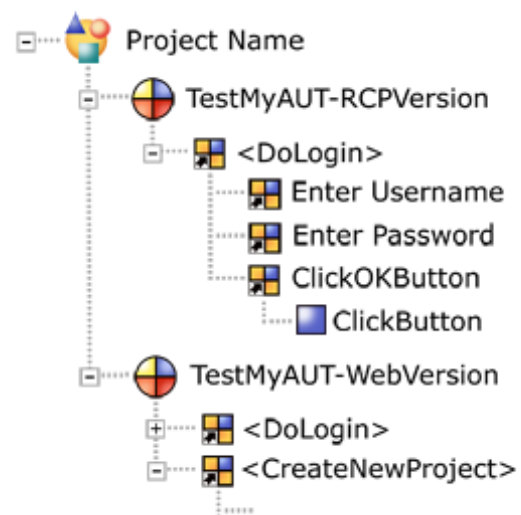
Of course, large tests can be created simply by referencing simple keywords again and again. Often though, small or large sequences of actions are suited to being made into a keyword. Small sequences such as "Select from Menu" and "Wait For Window to Open" can be combined in a keyword to open a certain dialog. Another good example is filling in a dialog. You could have a keyword that fills in the whole dialog with default data. This not only makes specification quicker, but also improves the maintainability of the tests.

Maintainability

As a general rule, any action or sequence of actions you need to do more than once should be a keyword. The keyword can be referenced whenever you need it. Any changes that need to be made to this keyword can then be made centrally, and the whole test is updated. Keywords remove redundancy from your tests and speed up test maintenance.

Flexibility

GUIDancer offers particular flexibility in keyword design. Data can be parameterised and separated from the test flow (data driven testing) or they can be encapsulated in the keyword so that the tester does not have to enter data each time. A combination of the two approaches can be used, and changes are easy to make should you want to. We also provide flexibility in terms of the user interface component tested by a keyword. You can choose to have your keyword executed on different (compatible) components in the application. For example, a keyword to select an item from a list can be designed to select *any* item from *any* list. A keyword to click in a component can be used to click *any* component. The advantage is that the same workflows in different dialogs or windows can use the same keywords.



Last, but not least

Keyword-driven testing with GUIDancer has two very specific advantages. Firstly, test automation is code-free. There are no programming tasks to do. Secondly, test creation is entirely independent of the application under test. You can start before it is even ready, and ensure that your tests are running as soon as the application is. Your tests can find errors as soon as the software is built, which improves quality and reduces development costs.