Welcome. In the next few minutes I would like to demonstrate to you how IBM Rational Functional Tester can be used to ensure the quality of web applications you may be deploying in your organization.

Today’s business environment is putting more and more demands on testing. Organizations need to reduce cost. Manual testing simply can't scale to the levels necessary to maintain high quality service to your customers and releasing low quality applications is simply not an option. Software solutions today are complex compositions of many components composed of legacy, homegrown, SOA and packaged applications. Utilizing unique tooling for each environment wastes valuable cash and time. We can't afford to reinvent the wheel. Quality must be part of the day to day culture of every team member. This drives the need for better collaboration and the facilitation of reuse to shorten test cycles. Functional test automation is one of the components of that quality management culture we will focus on in this video.

Of the many capabilities Rational Functional Tester provides, we will explore the basic workflow of creating an automated test by simply recording what you do manually, enhancing that test through the use of easy, intuitive user interfaces, executing an enhanced test, and exploring the execution results.

Let's get started.

IBM Rational Functional Tester is built using the open source Eclipse platform as its foundation. That means the user interface will feel very familiar to everyone on your team who has worked with Eclipse based tools already. Your task today is to construct an automated test that verifies your customers are able to edit fields on their profile records on your online banking application.

You will start by having Rational Functional Tester record your interactions as you make a profile change manually. The recorder engages, the Rational Functional Tester workbench minimizes itself, and the recording toolbar appears providing access to several wizards and tools that enable you to insert many advanced operations into your test while recording.

First you start your online banking application in the browser and begin to login as the user “Julie Brown”. Rational Functional Testers recorder is capturing every interaction you have with the application. That's great but you already know you are going to want to run this test again with different sample users. The recording toolbar enables you to designate fields for data driven testing. That means the data entered in the fields is separated from the steps of the test enabling you to add more datasets at any time.

You continue you scenario by clicking “Log In”. Your application proceeds to the customer details page.

Up to this point Rational Functional Tester has been watching you navigate the application but it hasn't really been told to test anything. Here you want to verify that the

right welcome message appears at the top of the screen so you use the recording toolbar to insert a verification point on the name. You associate the expected name to another variable in your dataset to make your test as flexible as possible.

Now you are ready to update Julies profile by clicking the “Update Profile” button which takes you to the customer profile page. You make some basic modifications such as changing the postal code for Julies address and save the change. You ensure the change is made properly by inserting another verification point on the modified data on the customer information page. You conclude the recording by logging out of the banking application, closing the browser and stopping the Rational Functional Tester recorder.

The first thing you notice back in the Rational Functional Tester workbench is the natural language test on the left. Statements like “Type jbrown” and “Click Log Insubmit” provide an easy to read test flow. You also notice as you click on the steps in the test, the screen visual on the right is also updated to reflect the content of the step. Screenshots make it very easy to understand exactly where you are in the test.

These are not simply screenshots though. The specific control related to each step in the test is highlighted on the image with a blue box. Furthermore, the other controls on the page are active. You can use a context menu to insert verification points, data driven commands and even action commands right from the visual image. Here you will insert a command to set the text of your User Id field. The ability to insert commands and verification points into a test without even executing the application is a huge time saver.

Functional Tester provides a number of other editing capabilities as well, such as disabling or deleting actions. For the power users you can even insert Java™ code snippets or calls to methods in supporting Java™ libraries to do whatever you might need.

Remember those data driven commands you inserted? Where do they get their data? Well, Rational Functional Tester recorded the values you entered as a start to your datapool. You can add additional data or even import extensive data sets from other sources such as databases or spreadsheets. Here you will enter one additional record.

You have one last enhancement to make to the test. Development has just delivered version two of your online application and they want you to run your test on it as soon as possible. You have already setup a shortcut to version two in Rational Functional Tester, so you make a quick change and you are ready to execute your test.

Executing the test is simply a matter of clicking the run button, naming the log file where results will be stored and telling Rational Functional Tester to run the test for each row in the datapool. Now you sit back and let Rational Functional Tester do the work.

Notice how version two looks much different. The objects are visually different and are

placed differently on the page. Application changes are typically the greatest weakness of an automated testing tool. But notice how Rational Functional Tester is able to see past the cosmetic and placement changes easily and play back without error.

One change does give Rational Functional Tester reason to pause. The “Update Profile” button has been changed to “Change Profile”. If the label were the only change, that would not have impacted Rational Functional Tester. However, the developers have changed not only the label but the programmatic name as well as other properties. Functional Tester’s patented ScriptAssure™ technology recognizes the “Change Profile” button as a relatively close match but it is waiting to see if the “Update Profile” button it expects will appear within its timeout settings. Eventually ScriptAssure™ assumes the “Change Profile” button has replaced the “Update Profile” button and uses it. A warning will be noted in the log but Rational Functional Tester is able to continue test playback. This is a powerful feature that can reduce script maintenance and increase your test teams’ efficiency.

When playback finishes, Rational Functional Tester presents the test result in one of several test log options. You have chosen the web based log utilizing Dojo technology. Notice there are some warnings and some failures. These are the warnings generated by ScriptAssure™ when it made the decision to use the “Change Profile” button when it couldn’t find the “Submit Profile” button. If this is a permanent change in the application you may want to either raise the threshold or update the Rational Functional Tester centralized object map to accommodate the change.

The errors are potentially more interesting. The verification point comparator indicates that although the welcome names in version one was the first name of the user, version two is using the last name. That seems rather impersonal. You have found a regression defect which is exactly why you use Rational Functional Tester in the first place. You will submit a defect work item to have development look into that.

I hope this brief video has given you an introduction to how you can use Rational Functional Tester to ensure the functional quality of your software applications. Keep in mind that everything that you saw today was done with no coding or scripting. This means your domain experts can stay domain experts without becoming programmers to test applications.

Also remember that functional verification is one part of the full line of quality management solutions offered by IBM Rational. Visit us soon on ibm.com.