$RCSfile: header.mm,v $ $Revision: 1.1.2.1 $ $Date: 1994/02/03 22:41:22 $ # $RCSfile: preface.sm,v $ $Revision: 1.1.2.1 $ $Date: 1994/02/03 22:42:10 $

Preface

Audience

This book is written with two audiences in mind:

- Testers who wish to use QATS as is to test OSF/Motif™, perhaps to verify vendor's implementations
- Testers who wish to automate testing of OSF/Motif-based applications by linking with the automation libraries

The task is straightforward for users in the first category, since it mirrors the Quality Assurance activities performed by test engineers at OSF prior to the release of OSF/Motif. In fact, the visual files shipped with QATS are the exact visuals used by OSF during the final tests of OSF/Motif Version 1.2.

Users in the second category are more likely to break new ground. The automation test code thusfar has been used only with the small test clients shipped as part of QATS. While we believe that the automation libraries can be used effectively to test real OSF/Motif applications, this is as yet unproven.

Applicability

Currently, QATS is shipped bundled with the OSF/Motif Validation Test Suite (MVTS). Used in conjunction, these two test suites provide a powerful and thorough test for OSF/Motif. Unlike MVTS, QATS is unsupported code, shipped "as is" to MVTS customers.

All of the QATS automation code was developed in-house at OSF.

The QATS test suite is not yet fully automated. In other words, not all directories contain scripts, and visual files have not been recorded for all directories. Those directories which have been automated are listed in the file `$TOP/tests/General/dirsAuto` on the QATS tape, where `$TOP` corresponds to the root of your QATS tree.

The next section contains the details of limitations in the current implementation which may necessitate modifications to your application, or to the automation technology, before automated testing of your OSF/Motif-based client can begin.

Purpose

The Quality Assurance Test Suite (QATS) is made up of several parts:

- Approximately 400 separate tests of OSF/Motif functionality
- Libraries of convenience routines used by these tests
- Libraries that support test automation
- Tools to help with test interpretation

The automation technology is the most significant new feature of the OSF/Motif Version 1.2 QATS. This package combines a comprehensive scripting language and input-synthesis extension with a platform-independent visual comparison algorithm. Together, it allows the test engineer to

- Write scripts which accurately reflect the actions of a human tester
- Record and compare visual appearance between different versions of any test, with great precision
- Automate existing tests with minimal or no modifications to the test code itself

Document Usage

This document is organized into the following sections:

- This **Preface**.
- **Chapter One**, which details the steps required to build the libraries, and to design and build automatable tests
- **Chapter Two**, which describes how to write automated test instruction and script files
- **Chapter Three**, describing the automated test run environment
- **Chapter Four**, which gives pointers on debugging problems discovered during automated testing
- **Chapter Five**, which describes some additional features of QATS and contains technical reference material.

- **Chapter Six**, which contains reference pages for the automation scripting language commands.

Related Documents

Users of this book may find the OSF/Motif documentation set useful, particularly the **OSF/Motif Programmer's Reference**, **OSF/Motif Programmer's Guide**, and **OSF/Motif Style Guide**.

Typographic and Keying Conventions

This document uses the following typographic conventions:

Bold — **Bold** words or characters represent system elements that you must use literally, such as commands, flags, and pathnames. **Bold** words also indicate the first use of a term included in the glossary.

Italic — *Italic* words or characters represent variable values that you must supply.

`Constant width` — Examples and information that the system displays appear in `constant width` typeface.

[ ] — Brackets enclose optional items in format and syntax descriptions.

{ } — Braces enclose a list from which you must choose an item in format and syntax descriptions.

| — A vertical bar separates items in a list of choices.

< > — Angle brackets enclose the name of a key on the keyboard.

... — Horizontal ellipsis points indicate that you can repeat the preceding item one or more times. Vertical ellipsis points indicate that you can repeat the preceding item one or more times.

This document uses the following keying conventions:

<Ctrl-*x*> or ^*x* — The notation **<Ctrl-*x*>** or ^*x* followed by the name of a key indicates a control character sequence. For example, **<Ctrl-c>** means that you hold down the control key while pressing **<c>**.

<Return> — The notation **<Return>** refers to the key on your terminal or workstation that is labeled with the word Return or Enter, or with a left arrow.

Entering commands — When instructed to *enter* a command, type the command name and then press **<Return>**. For example, the instruction "Enter the **ls** command" means that you type the **ls** command and then press **<Return>** (enter =

type command + press **<Return>**).

Componenents of the OSF/Motif user interface are represented by capital letters on each major word of the component, such as PushButton.

Problem Reporting

The Quality Assurance Test Suite provides a sophisticated test automation technology which, we hope, will be useful to all QATS licensees. The official status of the QATS is "unsupported", but we encourage you to report problems you find in the code or the documentation, and advise us of ways you find to extend the usefulness of the QATS.

Several bug fixes have been made to QATS since the release of OSF/Motif Version 1.2. These will be made available to MVTS/QATS support customers at a future time. Other future directions we are considering for QATS include:

- Complete automation of all directories which can be automated.
- Performance improvements.
- Support for currently unsupported OSF/Motif functionality, such as Tear-off menus, uil.
- Refinement of visual debugging tools.
- Removal of current restrictions, such as the requirement for `XmCreate` routines.
- Development of reporting tools, front ends.

All licensees should feel free to discuss the QATS on the **motif-talk** or **vts-talk** mailing lists, and to submit defects to **motif-defect@osf.org**. To find out about access to these mailing lists, contact **motif-request@osf.org**.

Companies who plan to make significant improvements, extensions or modifications to QATS, including any of the above possible futures, might consider contributing back their code to OSF for inclusion in future releases of QATS. To discuss joint development efforts, please contact Libby Hanna, QA/VTS Manager (**libby@osf.org** or (617) 621-8820).

OSF members can be actively involved priority-setting for future work. Much of this guidance takes place in the UEC (User Environment Componenet) SIG (Special Interest Group) for OSF/Motif. Contact your company's OSF representative to get more information about involvement in UEC SIG activities.