Smoke-Test

Date: 17.01.2024 CL-Vers.: 23.3c

Tester: Rolf

Build: 23.3.x (28yyy)

Hide u colum						Hide unn columns
115m	115m					
10	11	Test	Deal Braker	Expectation	Automation file path and description	Automation (Win)
		Preparation			The **SmokeTestSuite.cs** class, located in the **Test Automation Core\test\resources\test-suites** directory of the repository, is designed to execute this suite of smoke tests for ATLAS.ti Win. Running this test class will run the tests inside it sequentially. 1)Right-Click **SmokeTestSuite.cs** 2) Click on Run Tests.	
☐	☐	Find and download the test build (and install it)	☐		**SmokeTestSuite.cs => downloadRC()** • This method handles downloading Release Candidates before running tests. It uses a condition based on the AtlasVariables to determine if an update is necessary and then runs the update test. The existing installed version will be uninstalled automatically. The file will then be automatically downloaded to the Downloads folder. **However, due to a technical issue with the installer, manual installation will be necessary.**	✓
☐	☐	Create a folder *VUT_SmokeTest* on the test computer.	☐		**SmokeTestSuite.cs => initTestData()** • The \`initTestData()\` method in the \`SmokeTestSuite.cs\` class plays a pivotal role in preparing the test environment for smoke testing. Specifically designed to initialize data essential for the execution of smoke tests, this method is automatically invoked whenever a tester initiates the smoke test suite or any individual test within \`SmokeTestSuite.cs\`.	✓
☐	☐	Copy the Win folder in Smoke Test data in from oneDrive 👥 to test computer and unzip *Yanik/SmokeTestLibraryWin(Yanik).zip*	☐		**SmokeTestSuite.cs => initTestData()** All Smoke Test Libraries will be extracted automatically to *VUT_SmokeTest* folder	✓
☐	☐	Copy the following projects in C&H all versions from oneDrive 👥 to the *VUT_SmokeTest* folder: • Win Current Release • Win Previous Major • Mac Current Release • Mac Previous Major	☐		Projects will now be imported directly using their OneDrive file paths, eliminating the need to copy them into the VUT_SmokeTest folder. The file paths are stored at this class. **test\resources\test-data\onedrive\projects\CHProjects.cs**	✓
		Support Features			**SmokeTestSuite.cs => SupportTests()** . Running this individual test method will trigger all “support features” test cases sequentially	✓
☐	☐	Check for Updates (inside ATLAS.ti)	☐	“You are running the latest Version of ATLAS.ti”	**Test Automation Core\test\main\tests\smoketests\support\InternalUpdater.cs**	✓
☐	☐	Feedback & Help -> Live Chat Send question “ATLAS.ti QA test. Please ignore this.”	☐	Close this support request in help scout. (Ask Rolf to do so)	**Test Automation Core\test\main\tests\smoketests\support\LiveChat.cs**	✓
☐	☐	Help > Send System Report Provide your email address!	☐	email reply may take several hours	**Test-Automation-ATLAS.ti-.net-Core-\Test Automation Core\test\main\tests\smoketests\support\SendSystemReport.cs**	✓
☐	☐	Help > Send Feedback ... Provide message with up to 2000 characters.	☐	Review feedback in Application Insights (Ask Rolf to do so)	**Test Automation Core\test\main\tests\smoketests\support\SendFeedBack.cs**	✓
		Crash Test				
☐	☐	1. Send Crash report 1. Start ATLAS.ti in debug mode 2. Open a project, go to Developer > Raise Exception 3. relaunch ATLAS.ti and send Report, provide your email 2. In AppCenter search for **generateTestCrash** 3. Set status to “Ignored”	☐	• After re-launch, crash reporter opens • Report appears in AppCenter	Set status to ignored manually. This is the file path of the test case: **Test Automation Core\test\main\tests\smoketests\support\CrashTest.cs**	✓
☐	☐	Copy and unzip MigrationTest library from oneDrive 👥 (e.g. Testing Stuff/Test Data/Test Libraries/Smoke Test Yanik/SmokeTestLibraryWin(Yanik).zip)	☐	**Do we need a new Library?**	Done through: **SmokeTestSuite.cs => initTestData()** • Initialises test data specific to smoke testing.	✓
☐	☐	**Smoke Test B** – Run ATLAS.ti, migrate existing Library	☐		**SmokeTestSuite.cs => OpenYanikLibTest()**	✓
					All BackUp tests below will be executed with this test method **SmokeTestSuite.cs => BackUpTests()**	
☐	☐	Open the Backup & Restore Tool while ATLAS.ti is running	☐	Warning - denied	**Test Automation Core\test\main\tests\smoketests\backuptests\BackupTest1.cs**	✓
☐	☐	Close ATLAS.ti, then create a backup (or: Open the Backup & Restore Tool while ATLAS.ti is NOT running)	☐	Restore tool opens		✓
☐	☐	Create Backup	☐	Success notification - atlbak9 file is generated	**Test Automation Core\test\main\tests\smoketests\backuptests\BackupTest2.cs**	✓
☐	☐	Open ATLAS.ti, delete a project	☐	project lists only 2 projects	**Test Automation Core\test\main\tests\smoketests\backuptests\BackupTest3.cs**	✓

Hide u colum						Hide unn columns
115m	115m					
10	11	Test	Deal Braker	Expectation	Automation file path and description	Automation (Win)
☐	☐	Restore a Backup, then open ATLAS.ti	☐	Success notification - deleted project is back	Restore BackUp=> **Test Automation Core\test\main\tests\smoketests\backuptests\BackupTest4.cs** Open Restored Project=> **Test Automation Core\test\main\tests\smoketests\backuptests\BackupTest5.cs**	✓
						✓
☐	☐	1. Copy and unzip test library from oneDrive 👥 2. 05 - Smoke Test Library Containing C&H - Win.zip, 3. Open it via Switch Library wizard 4. Open the C&H project	✓		**SmokeTestSuite.cs => OpenYanikLibTest()**	✓
		In VUT import projects FROM released Versions 23			**SmokeTestSuite.cs => OtherCHImportTests()**	
☐	☐	Import an **AtlProj23** exported with current Win Release	☐	Migrate Project	**Test Automation Core\test\main\tests\smoketests\migrationtests\chprojects\otherversions\ImportCHAtlProj.cs. Purpose of this class:** • Testing Different Versions of Projects: The class focuses on testing the import functionality for various versions of CH ATLAS.ti projects, ensuring compatibility and correctness across different versions. • Data-Driven Testing: By using TestCaseData and TestCaseSource, the class implements a data-driven approach, allowing multiple test scenarios to be executed with different data sets.	✓
☐	☐	Import a **QDPX** exported with current Win Release	☐	Migrate Project	**Test Automation Core\test\main\tests\smoketests\migrationtests\chprojects\otherversions\ImportCHQDPX.cs**	✓
☐	☐	Import an **AtlProj23** exported with current Mac Release	☐	Migrate Project	**Test Automation Core\test\main\tests\smoketests\migrationtests\chprojects\otherversions\ImportCHAtlProj.cs**	✓
☐	☐	Import a **QDPX** exported with current Mac Release	☐	Migrate Project	**Test Automation Core\test\main\tests\smoketests\migrationtests\chprojects\otherversions\ImportCHQDPX.cs**	✓
		In VUT import projects FROM previous major v. 22				
☐	☐	Import an **AtlProj22** exported with previous Win major	☐	Migrate Project	**Test Automation Core\test\main\tests\smoketests\migrationtests\chprojects\otherversions\ImportCHAtlProj.cs**	
☐	☐	Import a **QDPX** exported with previous Win major	☐	Migrate Project	**Test Automation Core\test\main\tests\smoketests\migrationtests\chprojects\otherversions\ImportCHQDPX.cs**	✓
☐	☐	Import an **AtlProj22** exported with previous Mac major	☐	Migrate Project	**Test Automation Core\test\main\tests\smoketests\migrationtests\chprojects\otherversions\ImportCHAtlProj.cs**	✓
☐	☐	Import a **QDPX** exported with previous Mac major	☐	Migrate Project	**Test Automation Core\test\main\tests\smoketests\migrationtests\chprojects\otherversions\ImportCHQDPX.cs**	✓
☐	☐	**Quit VUT**	☐		**Test Automation Core\test\main\tests\BaseTest.cs => cleanUp()** The cleanUp method will be triggered automatically after each test case located in a child test class of the BaseTest.cs class. It serves as a teardown procedure for test cases. The cleanUp method is crucial for maintaining a clean testing environment. It ensures that after each test execution: • Necessary artifacts like screenshots are saved for future reference. • The system is returned to a stable state by terminating any processes started during the test. This method helps in preventing side-effects from one test affecting subsequent tests, a critical aspect in automated testing for consistency and reliability of test results.	✓