GUIDEWIRE
SUREPATH

Designated Function Tests for PolicyCenter GT: UI – Detailed Design

Guidewire SurePath Reference Implementation for PolicyCenter

Prepared by Guidewire

Table of Contents

Introduction

Overview

This package contains Guidewire Testing User Interface (GT: UI) Test Scenarios that provide coverage for PolicyCenter Designated Functions, which should be met as part of the Guidewire Cloud testing standards. These test scenarios are behavior tests written in a common language using Cucumber's Gherkin syntax. The tests use Personal Auto LOB as a representative example, and the tests provide guidance on the approach and patterns compliant with the Guidewire Cloud testing standards. The tests can be used as a reference implementation for other lines of business to meet designated function coverage.

This document provides instructions for using this test content for Guidewire PolicyCenter Cloud Implementation projects.

Intended Audience

This document is intended for developers and test automation resources who will be implementing, deploying, or extending this reference implementation. It is also intended for implementation business analysts, subject matter experts, and quality assurance analysts who will be using, evaluating, or testing this reference implementation. PMO colleagues also may gain value from this document.

Please read the latest version of the GT: UI product documentation, as these documents will guide you in best practices for planning, developing, and executing Functional UI tests.

Applicable Product Versions

These reference tests are compatible with both the Hakuba and Garmisch releases of Guidewire Testing Framework (GT: Framework 2023.02 and 2023.06) to be run against either the Hakuba or Garmisch release of PolicyCenter Cloud.

Terms of Use

We are providing this document and the corresponding Content Module and making it available to you to help accelerate your implementation and provide a standardized configuration pattern for this package. This document and its related content are provided "as-is," which means that we do not offer you any assurances with respect to them. You will be solely responsible for any changes made to your configuration as a result of implementing any of the Content materials. In addition, any mention of known issues or planned future enhancements in this document does not represent formal commitments on product direction by Guidewire.

You also understand that we own the intellectual property rights to any documentation or code that we make available to you. You must not use them in any way that would adversely affect our rights under applicable law.

Installation Guide

This Content package can be installed by extracting the contained files referenced in this section and placing them in specific directories in your GT: UI installation. This section provides specific guidance on modifications made to existing files so that your installation does not inadvertently revert any other code written in those files by your implementation.

The installation procedure assumes that the GT Framework has been installed. For GT Framework installation procedures, refer to the “Install Guidewire Testing Framework” section in *Guidewire Testing Framework Installation*.

Zip File Extraction

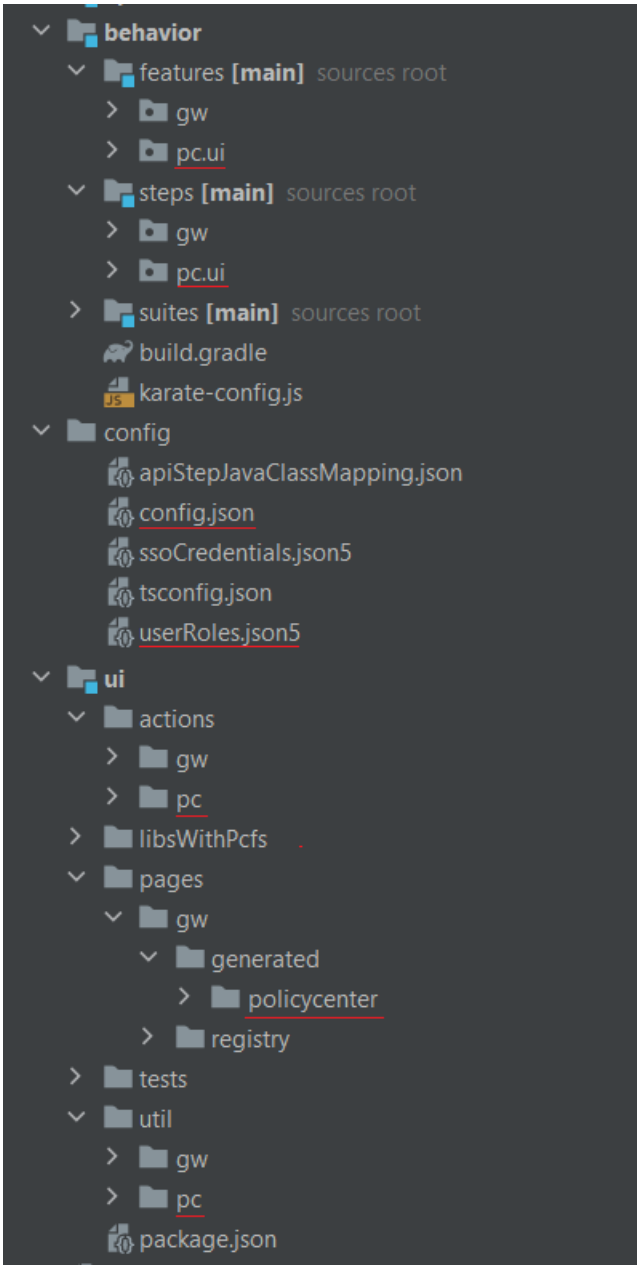
Extract the contents of the Zip file into the `gt-framework` directory. Many files will be extracted. The contents of the Zip file are intended to be an overlay on top of the files in the `ui`, `behavior`, and `config` subdirectories. When prompted, select **Overwrite** to replace the corresponding files.

Run Ignite to Create Page Models

The Content relies on the Page Models created by running Ignite being in the `ui/pages/gw/generated/policycenter` directory. Follow the instructions in *User Interface Testing* available in the product documentation. Refer to the section "Ignite: Automatic page object generation." This section guides you through the steps of creating a PCF JAR file and running the Ignite command.

New Files – GT: UI Test Content for PolicyCenter Designated Functions

Expanding the Zip file places the entire `SurePath` directory's contents in their correct locations as follows:



Note the `policycenter` directory under `pages/gw/generated` was created in the previous step when Ignite was executed.

Modified Files – GT: UI Test Content for PolicyCenter Designated Functions

Two files must be updated to work with the SurePath content. These files are:

- `config.json`
- `userRoles.json5`

Both files reside in the `config` directory. You can copy the files from the expanded Zip file or just update the relevant lines. For the `config.json` file, the lines that you need to be update are:

```
{
  "PC_URL": "http://localhost:8180/pc/PolicyCenter.do",
  "CC_URL": "http://localhost:8080/cc/ClaimCenter.do",
  "BC_URL": "http://localhost:8580/bc/BillingCenter.do",
  "CM_URL": "<Please enter ContactManager app URL>",
  "dev-test-location-pattern ": "ui/tests/**/*.js ",
  "feature_location_pattern": "behavior/features/**/*.feature",
  "glue_location_pattern": "behavior/steps/**/*.js",
  "allow_insecure_app_urls": "true"
}
```

- Update the first highlighted line to point to the running PolicyCenter application under test.
- Include the `"allow_insecure_app_urls"` line if running against a local version of Policy00000Center.

For the `userRoles.json5` file, you need only update the PolicyCenter section to include the following entries:

```
PC: [{
      userrole: "superuser",
      username: "su",
      password: "gw"
    },{
      userrole: "producer",
      username: "su",
      password: "gw"
    },{
      userrole: "underwriter",
      username: "aapplegate",
      password: "gw"
   }]
```

Setting Up PolicyCenter

This content uses the base configuration of PolicyCenter with a few adjustments. You can run tests against a PolicyCenter application that is either running locally or running on the cloud platform. For simplicity, the following instructions assume that PolicyCenter is running on the cloud platform.

Loading Sample Data

The test content expects PolicyCenter to be running with Sample Data loaded. Log in to the running PolicyCenter instance using the su/gw credentials, press Alt-Shift-T, and then select **Internal Tools – PC sample Data** from the tabs on top. Click **Load** for the small data set.

Executing Tests

In the terminal window of your IDE run the command

```
npm run gt-ui-tests -- --feature= behavior/features/pc/ui/*.feature
    --cucumberTags='@DesignatedFunction'
```

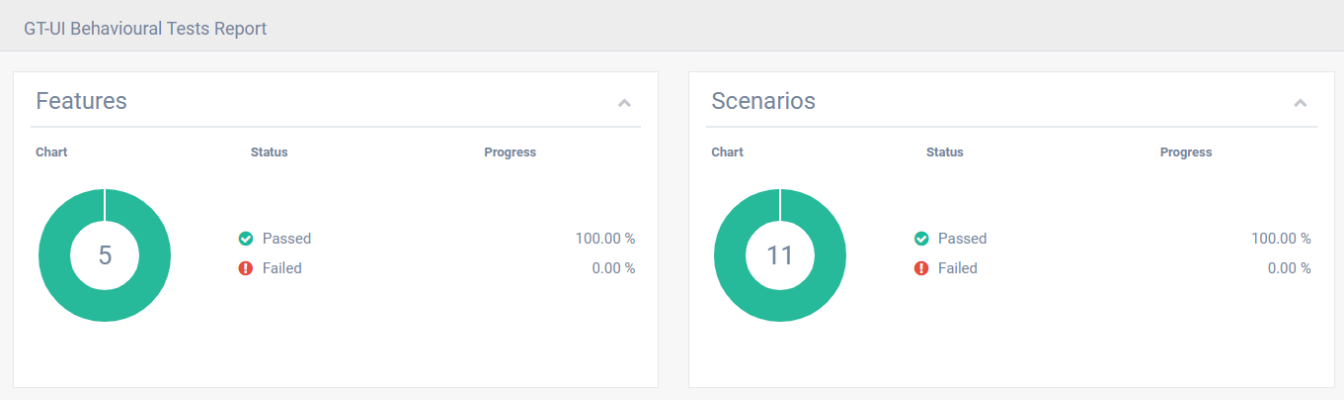
This command runs all the designated function tests for PolicyCenter only.

Viewing Test Results

To view the results, run:

```
npm run report
```

The results should look similar to the following:



Accessibility Scans

Executing an Accessibility Scan

The Hakuba release of Guidewire Testing Framework includes an Early Access feature that runs an accessibility scan on page objects. The SurePath content includes an example of how the accessibility scan feature works. To see details for this feature, refer to the Guidewire product documentation and follow the links to *User Interface Accessibility Testing* in the Guidewire product documentation.

The basic concept comprises three steps.

1. Import `AxeScanWrapper` into the test you want the scan.
2. Call `AxeScanWrapper` on the initial call to the page object.
3. Add `-axeScan=true` to the npm command line when invoking your test suite.

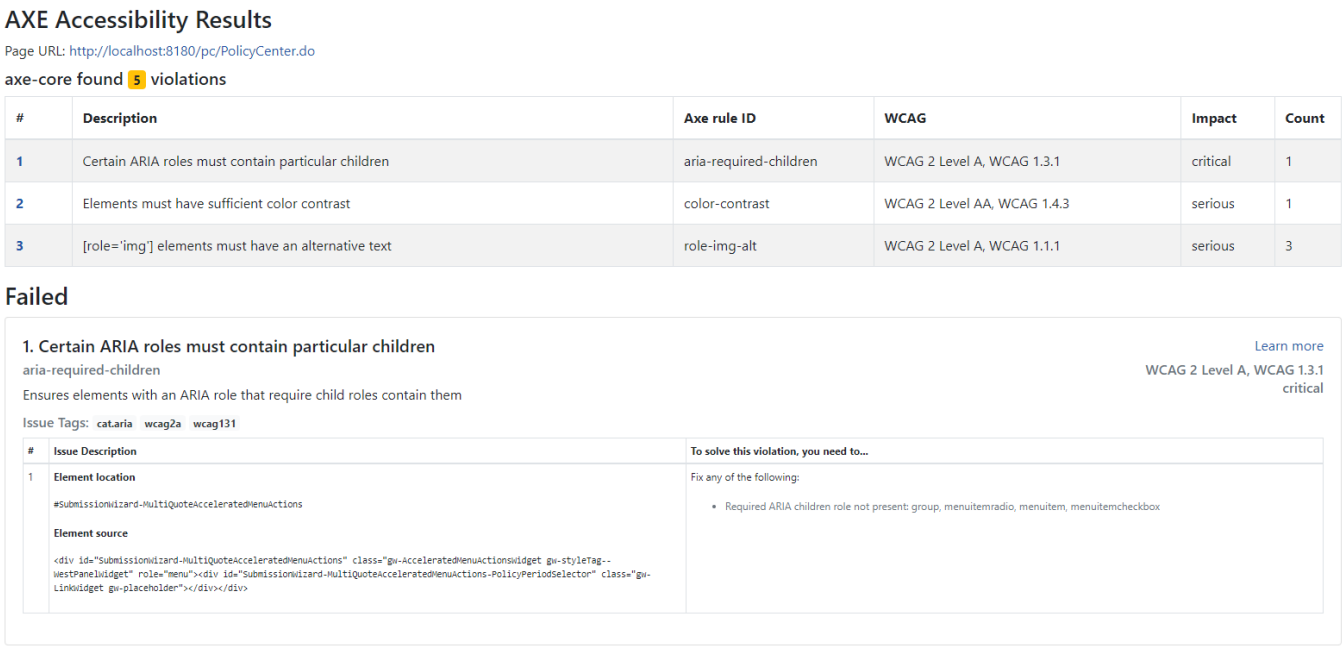
The example in the SurePath content is in the `PA-SimpleRate.feature`, on the `Forms inferred when quoting a Personal Auto submission` scenario. It is tagged with `@AccessibiltyScanning` for easy execution.

To run this content, reference the tag and include `-axeScan=true` on the run command, as follows:

```
npm run gt-ui-tests -- '--feature=behavior/features/pc/ui/*.feature'
    --cucumberTags='@AccessibiltyScanning' --axeScan=true
```

Viewing the Accessibility Scan Report

After it is executed, the accessibility scan report is generated and placed in the `reports/uiaxetests` directory. In this case, the report is named ClaimSummaryResults.html. Open the report in a browser. A typical report looks similar to the following:



AXE Accessibility Results

Page URL: http://localhost:8180/pc/PolicyCenter.do

axe-core found 5 violations

#	Description	Axe rule ID	WCAG	Impact	Count
1	Certain ARIA roles must contain particular children	aria-required-children	WCAG 2 Level A, WCAG 1.3.1	critical	1
2	Elements must have sufficient color contrast	color-contrast	WCAG 2 Level AA, WCAG 1.4.3	serious	1
3	[role='img'] elements must have an alternative text	role-img-alt	WCAG 2 Level A, WCAG 1.1.1	serious	3

Failed

1. Certain ARIA roles must contain particular children

aria-required-children

Ensures elements with an ARIA role that require child roles contain them

Learn more

WCAG 2 Level A, WCAG 1.3.1

critical

Issue Tags: cat.aria wcag2a wcag131

#	Issue Description	To solve this violation, you need to...
1	**Element location** #SubmissionWizard-MultiQuoteAcceleratedMenuActions **Element source** 	Fix any of the following: • Required ARIA children role not present: group, menuitemradio, menuitem, menuitemcheckbox

Functional Design

Test Scenarios

All GT: UI Test Scenarios shipped with this package reside in the `behavior/features/pc/ui` directory.

Tests for LOB-Agnostic Designated Functions have a `COM-` prefix. Tests for LOB-Specific Designated Functions have a `PA-` prefix. Personal Auto LOB is used as a representative example for LOB-Specific Designated Functions.

LOB-Agnostic Scenarios

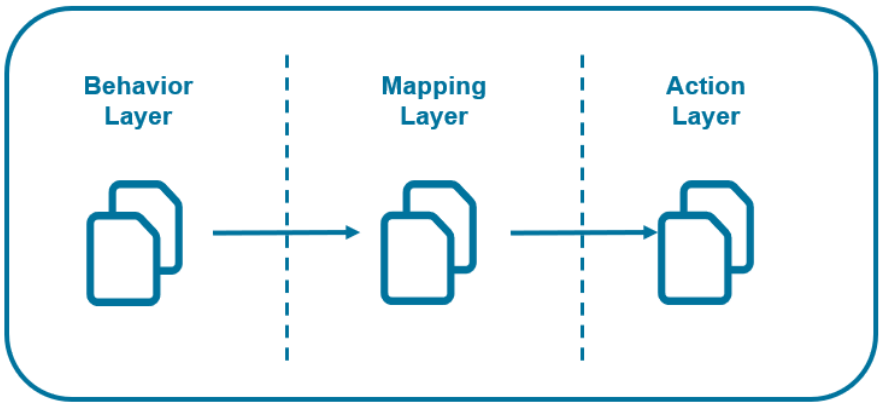
- `COM-AccountOverview.feature` - Tests the ability to view the overview of the account.
- `COM-ContactSearch.feature` - Contains scenarios that test the ability to search for various types of contacts using different identifiers.

LOB-Specific Scenarios

- `PA-Billing.feature` - Contains scenarios that test for the creation of billing events.
- `PA-FormsInference.feature` - Contains scenarios that test the inference of forms during the process of a submission.
- `PA-SimpleRate.feature` – Tests the rating of a Personal Auto submission.

Technical Design

The behavior tests shipped with this package use the following design for easier maintenance and to be compliant with cloud testing standards.



Behavior Layer

This layer contains Cucumber test features expressed in common language using Gherkin Syntax and their associated step definitions. All feature files are stored in the `behavior/features/pc/ui` directory. This directory contains the files mentioned in “Test Scenarios.”

Mapping Layer

The step definitions map the lines of the provided feature files to the scenarios. All step definitions are located in the `behavior/steps/pc/ui` directory. This directory contains the following files:

- `AccountSteps.js` – step definitions for account actions.
- `BillingSteps.js` – step definitions for billing actions.
- `ContactSteps.js` – step definitions for contact actions.
- `FormSteps.js` – step definitions for form actions.
- `JobSteps.js` – step definitions for job actions.
- `PolicySteps.js` – step definitions for policy actions.
- `PremiumSteps.js` – step definitions for rating actions.
- `SubmissionSteps.js` – step definitions for submission activities.

Action Layer

The scenarios contain the glue code that interacts with the application. All the scenarios are located in file in the `ui/actions/pc/scenarios` directory. This directory contains the following items:

- ScenarioPages
  - `Account/*`
    - Page models for updated/added page objects around the account entity.
  - `SubmissionWizard`
    - Page models for updated/added page objects around the Submission Wizard.
      - `AccountScenario.js` – Steps for working with Accounts.
      - `ContactScenario.js` – Steps for working with Contacts.
      - `SubmissionScenario.js` – Steps for working with Submissions.

Utilities

The `ui/utils/pc` directory contains the following utilities:

- `world.js` – a file for passing data to and from the scenarios.

Document Information

Document History

Version Number	Revision Date	Summary of Changes
1	June 2021	Initial version
2	November 2021	Updated version for the Dobson release
3	June 2022	Updated version for the Elysian release (2022.05)
4	May 2023	Updated version for the Garmisch release (2023.02)
5	August 2023	Updated version for the Garmisch & Hakuba releases (2023.06 & 2023.02)