New list of improvement

June 13, 2022 17:38

List of improvement:

1. Tooling and Development
    a. Make the “Cucumberized”JUnit as a Maven package
2. User Experiences
    a. Use dedicated exceptions with error code and human readable error message to help developers determining the issues
3. Refactor and improvement
    a. JScenario code should be separated into several classes for better maintainability and testability. New separated classes include but are not limited to Jfeature, JScenario, JScenarioBuilder, and etc.
    b. Nested collection such as List of Map of List is not a best practice, instead data classes should be created for storing information.
    c. If a step has a number in the JScenaio description, the project always tries to find the method that has an int parameter. Relax this requirement and allow empty parameter step definition method matches this kind of step.
        i. For example, "I have 3 apples" will currently always be mapped to a step definition method with an int parameter, such as "void my_definition(int numberOfApples)”
    d. The name of the step definition method must be written in a fixed pattern that matches what is written in the JScenario description. Relax this requirement.
        i. For example, to match "I have an apple", the method name must be "i_have_an_apple()"
4. New features
    a. Support Tags [2], as well as allow running subset of scenarios
        i. (Optional, time-permitting) Supports tags on example tables which allows a scenario outline to have multiple example table candidate and enable one of them through tags
    b. Support comments in #, and """ doc string
    c. Support keyword “And”and “But”
    d. Support parameter types of double, BigDecimal, BigInteger
    e. Allow users providing their own instance of the step definition class, instead of relying on JScenario to create an instance of step definition through Java Reflection. This allows integration with one’s favourite dependency injection framework like Spring.
    f. Allow users providing multiple step definition class/instances for one scenario, like how Cucumber does it
    g. (Optional, time-permitting) Support Background [3], a “Scenario”that runs before all Scenarios
    h. (Optional, time-permitting) Support selected Hooks [4]such as BeforeAll, AfterAll, etc.
    i. (Optional, time-permitting) Support i18n [5]
    j. (Optional, time-permitting) Support custom data type conversion [6]

[1] Cucumber, "Behaviour-Driven Development - Cucumber Documents," SmartBear Software, [Online]. Available: https://cucumber.io/docs/bdd/.

[2] Cucumber, "Tags," SmartBear Software, [Online]. Available: https://cucumber.io/docs/cucumber/api/#tags.

[3] Cucumber, "Background," SmartBear Software, [Online]. Available: https://cucumber.io/docs/gherkin/reference/#background.

[4] Cucumber, "Hooks," SmartBear Software, [Online]. Available: https://cucumber.io/docs/cucumber/api/#hooks.

[5] Cucumber, "Spoken Languages," SmartBear Software, [Online]. Available: https://cucumber.io/docs/gherkin/reference/#spoken-languages.

[6] Cucumber, "Type Registry," Spoken Languages, [Online]. Available: https://cucumber.io/docs/cucumber/configuration/#type-registry.