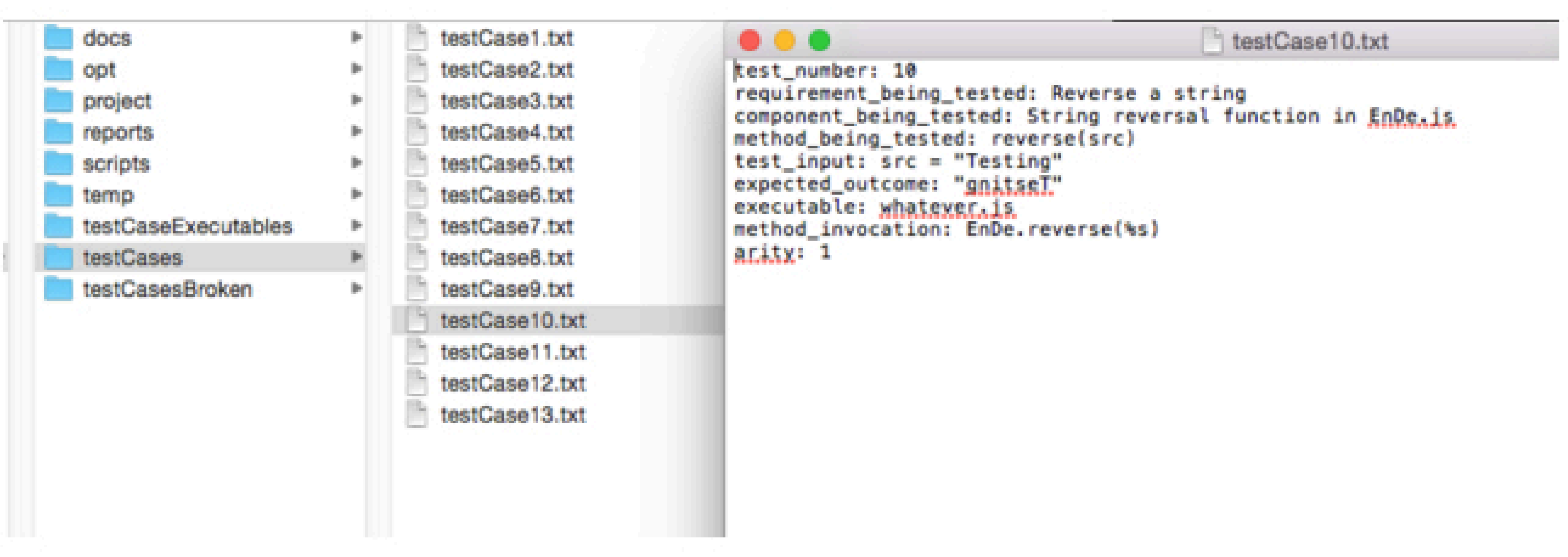
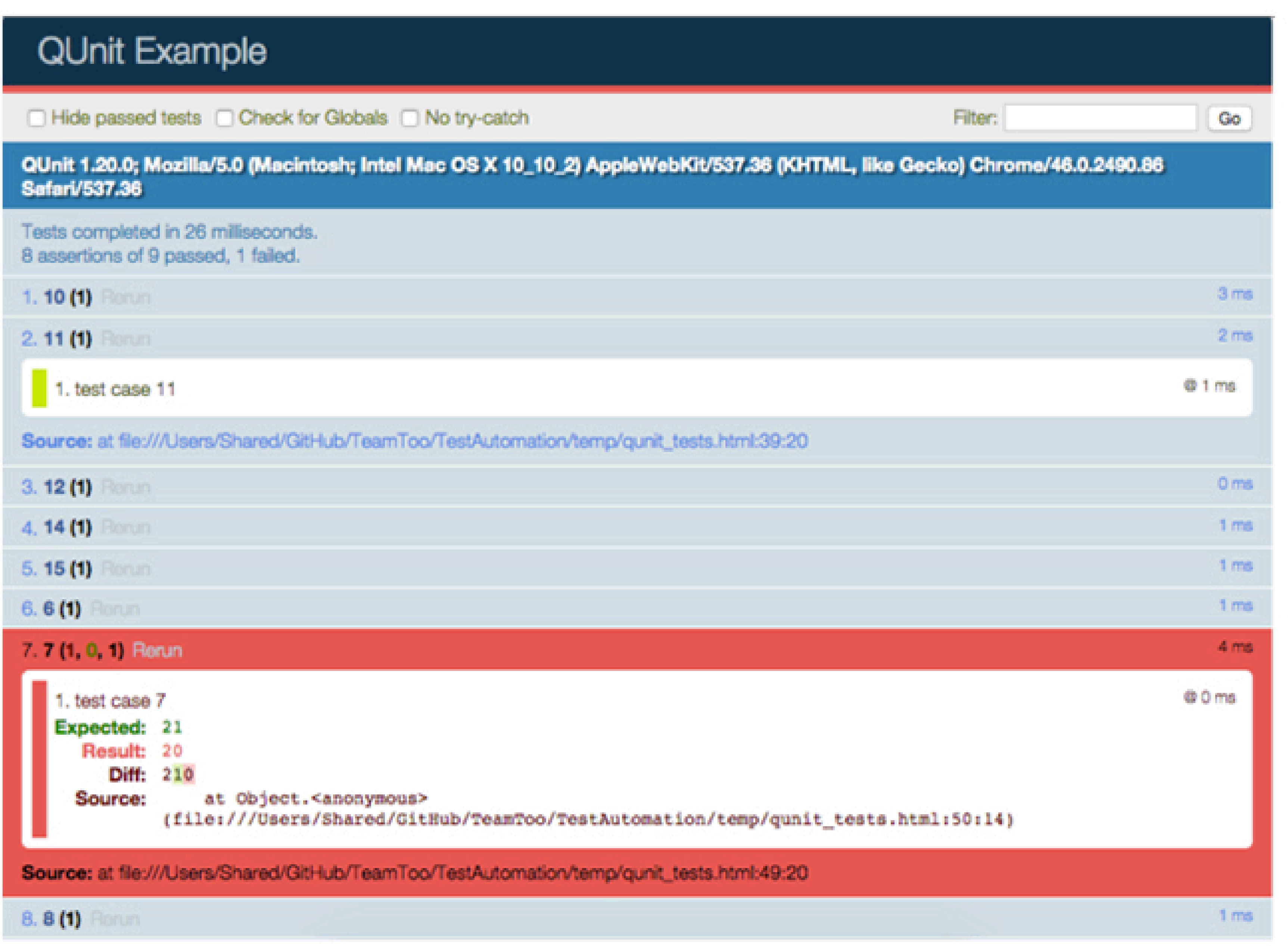
Automated Testing Framework: Adding New Logic for New Tests

Some changes were made in how our testing framework works since Chapter 3. Now, instead of using the names/numbers of the tests to run separate files in the testCaseExecutables directory, we run all tests using information in the .txt files found in the testCases directory.

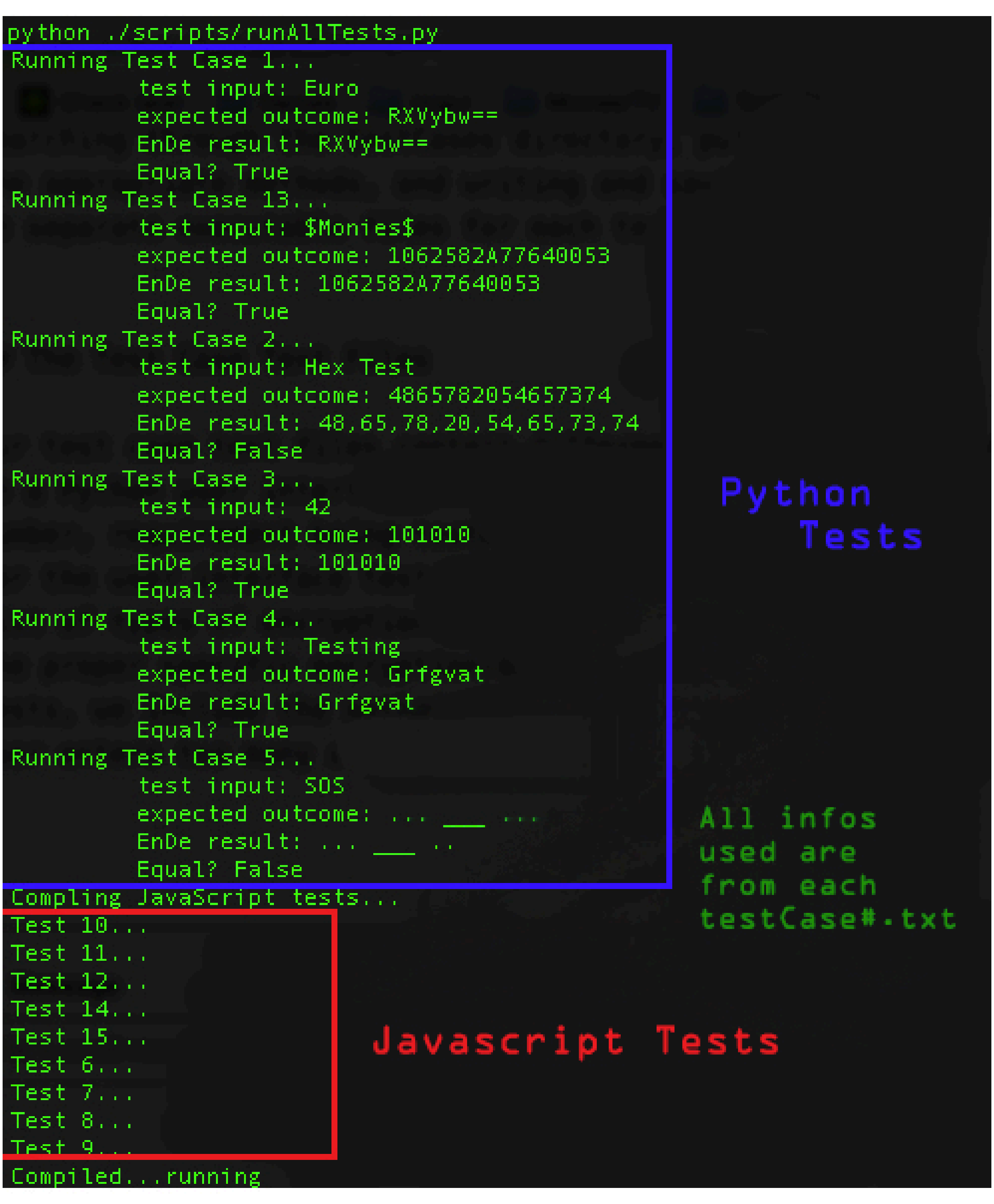


These files contain all information needed to execute the tests without needing repeated code in the testCaseExecutables files. We use two different types of cases in our testing: tests written in Python that use Selenium to simulate interactions in a browser window to test the user interface, and tests on the JavaScript code that EnDe is largely written in.

In runAllTests.py, we check whether we need to be running Python code to test the user interface or JavaScript code to test methods directly. For the JavaScript tests we used QUnit, which is a JavaScript unit testing framework. We were able to get the results of whether a test passed or failed using QUnit, then used those values in our report.

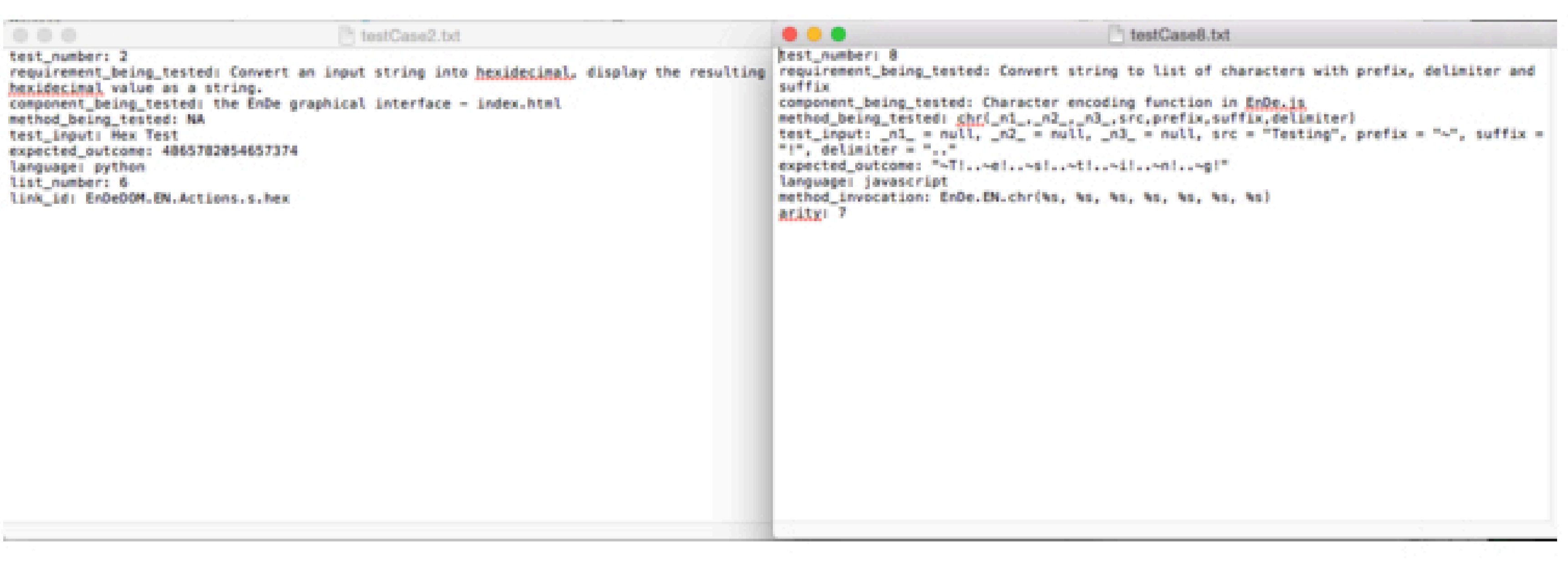


Almost all of the functionality for our automated testing framework is in runAllTests.py. This includes searching through the testCases directory, pulling out relevant information from the files and invoking the appropriate methods, and writing and saving the report. This was done mainly to avoid repeated code in separate executable files for each test case.



The Test Case Text Files

Our test case text files contain different information depending on whether the information will be used in a Python user interface test or a JavaScript test for functions. All test case files contain a test number, requirement being tested, component being tested, test input, and expected outcome (oracle). For the user interface tests, we also include a list number which allows us to locate which item in the list of types of encryption we need to click in the user interface and a link id that allows us to click the proper specific encryption/decryption method in a sub-menu under that type. For our JavaScript tests, we include the method being tested and how to invoke the method. We also include arity, so if a user enters too many arguments it will only accept the first ones up to the amount required.



Errors and Manual Checks

We did find some issues with using the test case files, however. One issue arose when implementing automated testing for values that were returned not as UTF-8 encoded characters. Of course, this means that the test did fail, but it would also cause the scripts to break, and it would cause issues with the files being written (ie: the results.html). Since this was causing the automated tests to fail, and scripts to be broken, this test had to be checked manually to verify the issue.

The culprit test case:

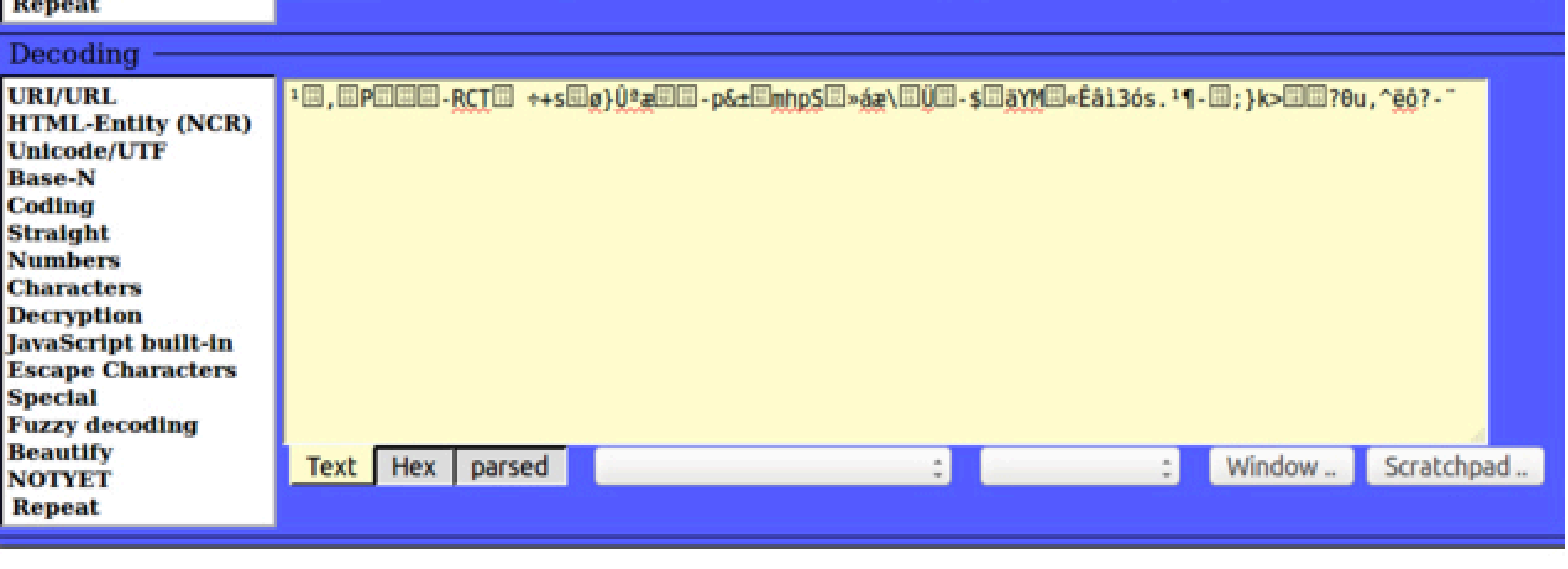
```
testCase10_broken.txt — Edited
test_number: DON'T USE THIS ONE
requirement_being_tested: AES-128 encoding - verifiying that output matches expected from web browser
component_being_tested: the EnDe graphical interface - index.html
method_being_tested: AES-128 encoding
test_input: heureca!
expected_outcome: \xa3\x98\x17\xc9\x26\x01\x00\x00\x2d\x7c\x4d\x3b\xfe\x1d\xc2\x01\x07
executable: testCase9.py
list_number: 9
link_id: EnDeDOM.EN.Actions.s.aes128

SERIOUSLY. DON'T USE IT. IT WILL BREAK EVERYTHING.
```

The error being thrown:

```
Running Test Case 10...
        test input: heureca!
        expected outcome: \xa3\x98\x17\xc9\x26\x01\x00\x00\x2d\x7c\x4d\x3b\xfe\x1d\xc2\x01\x07
        EnDe result: #µçùP-X,_<Ôðû
        Equal? False
Traceback (most recent call last):
  File "./scripts/runAllTests.py", line 129, in <module>
    testCaseData['expected_outcome'], val))
UnicodeEncodeError: 'ascii' codec can't encode characters in position 302-304: ordinal not in range(128)
```

Notice how the output recorded in the console only shows 'some' non-standard characters. Other characters aren't shown because the console can't display them.

Sample manual test from the GUI:



Notice in this one the non-standard characters that are shown as boxes. These are what is attempting to be added to the results.html file. The only reason they show up here as boxes is because that is what Firefox has implemented to catch non standard characters like these.

In order to prevent the error from popping up while testing, and causing issues in doing so, entering any value other than an numerical one will cause the testing to stop before the values are sent to test.

```
Traceback (most recent call last):
  File "./scripts/runAllTests.py", line 73, in <module>
    testCaseIndex = int(testCaseData['test_number'])
ValueError: invalid literal for int() with base 10: "DON'T FREAKING USE THIS ONE"
```