Using Data From Standard IO

This topic explains how you can have tests examine data which is sent to standard output or standard error streams. This is accomplished by interacting with the C++test Stream API, which is designed for using standard input/output streams. This API allows manipulation and content verification of streams.

Sections include:

Using the C++test Stream API

To have tests examine data which is sent to standard output or standard error streams:

1. To redirect the stream before test case execution, use one of the following functions:

   ```
   CppTest_StreamRedirect* CppTest_RedirectStdOutput()

   CppTest_StreamRedirect* CppTest_RedirectStdError()
   ```

   Both `CppTest_StreamRedirect* CppTest_RedirectStdOutput()` and `CppTest_StreamRedirect* CppTest_RedirectStdError()` return a pointer to a special internal C++test structure or NULL (in case of error). This pointer can be used later in other stream redirection functions, as illustrated in Example.
2. To compare the content of streams with a null-terminated string or a data buffer, use the following functions:

   ```
   CppTest_StreamCompare(CppTest_StreamRedirect* redirect, const char* value);

   CppTest_StreamNCompare(CppTest_StreamRedirect* redirect, const char* value, unsigned int size);
   ```

   Both functions return 0 if the value matches the stream value, an integer less than zero if the value is less than the stream value, or an integer greater than zero if the value is greater than the stream value.

   `CppTest_StreamCompare` behaves like strcmp.

   To get the whole data buffer from the stream using a function, use `char* CppTest_StreamReadData(CppTest_StreamRedirect* redirect, unsigned int* len);` This function returns a pointer to a buffer with data from the stream. The size of the buffer will be passed back in the `len` parameter. The buffer is allocated with the `malloc` function. You are responsible for freeing this buffer.
3. To set values for the input stream, use the following initialization functions:

   ```
   CppTest_StreamRedirect* CppTest_RedirectStdInput(const char* value);

   CppTest_StreamRedirect* CppTest_RedirectNStdInput(const char* value, unsigned int size)
   ```

   These functions set the input stream to the value of the supplied string (using N characters in the second case).
4. At the end of the test case, reset the C++test internal stream to the default state with

   ```
   void CppTest_StreamReset(CppTest_StreamRedirect* redirect);
   ```

Example

```
/* CPPTEST_TEST_CASE_BEGIN foo_test */
void TestSuite::foo_test()
{
  /* Pre-condition initialization */
  CppTest_StreamRedirect* output_stream = CppTest_RedirectStdOutput();

  /* Tested function call */   ::foo();

  /* Post-condition check */
  /* This assert validates that "Output string" is put by foo() on standard output. */
  CPPTEST_ASSERT(0 == CppTest_StreamCompare(output_stream, "Output string"));
  CppTest_StreamReset(output_stream);
}
/* CPPTEST_TEST_CASE_END foo_test */

/* CPPTEST_TEST_CASE_BEGIN bar_test */
void TestSuite_a_0::bar_test()
{
  /* Pre-condition initialization */
  CppTest_StreamRedirect* input_stream = CppTest_RedirectStdInput("Hello World!\n");

  /* Tested function call */
  /* bar() reads standard input and returns the string */
  std::string ret = ::bar();

  /* Post-condition check */
  CPPTEST_ASSERT(ret == "Hello World!");
  CppTest_StreamReset(input_stream);
}
/* CPPTEST_TEST_CASE_END bar_test */
```