# TestHelper

TestHelper to biblioteka pomocnicza do testowania deweloperskiego. W odróżnieniu od frameworków testowania jednostkowego, takich jak MS Test, Nunit, XUnit.net itp., jest przeznaczona nie tylko do wykonywania testów na zasadzie passed/failed, ale służy bardziej do śledzenia tego, co się dzieje w czasie wykonywania testów. Dzięki temu deweloper może się łatwiej zorientować w sposobie wykorzystania zewnętrznych komponentów, których nie może śledzić bezpośrednio, a także w działaniu skomplikowanych algorytmów, w których może zaszyć mechanizmy podglądu wyników pośrednich i cząstkowych.

Generalnie infrastrukturę komponentu TestHelper można podzielić na cztery obszary:

* TraceWriting – śledzenie przez zapisywanie danych, wyników i zdarzeń,
* FileComparing – porównywanie plików (zwłaszcza z wynikami),
* TestBuilding – tworzenie i uruchamianie testów,
* Others – pozostałe.

## TraceWriting

Infrastruktura TraceWriting jest zdefiniowana w oparciu o interfejsy:

* ITextWriter – definiuje operacje pisania tekstu łącznie z wcięciami tekstu.
* IConsoleWriter – rozszerza interfejs ITextWriter o możliwości pisania tekstu w kolorze na konsoli systemowej,
* ITraceTextWriter – rozszerza interfejs IConsoleTextWriter o możliwość równoczesnego pisania tekstu na konsolę, do klasy Trace oraz do pliku (strumienia) tekstowego,
* ITraceXmlWriter – rozszerza interfejs ITraceTextWriter o możliwość tworzenia struktury Xml,
* ITraceMonitor – umożliwia zbieranie powiadomień z monitorowanej aplikacji i ich zbiorcze przetwarzanie

oraz implementujące je klasy:

* BufferedTextWriter – abstrakcyjna klasa implementująca interfejs ITextWriter w oparciu o TextWriter przy pomocy bufora.
* ConsoleWriter – implementuje interfejs IConsoleWriter, umożliwia pisanie na konsolę z użyciem kolorów.
* TraceWriter – rozszerza klasę ConsoleWriter, implementuje interfejs ITextWriter, umożliwia jednoczesne pisanie tekstu na konsolę i do klasy Trace, kolor jest interpretowany przy wyjściu na konsolę,
* TraceTextWriter – rozszerza klasę TraceWriter. implementuje interfejs ITraceTextWriter, umożliwia jednoczesne pisanie tekstu na konsolę, do klasy Trace i do pliku/strumienia, kolor jest interpretowany przy wyjściu na konsolę,
* TraceXmlWriter – rozszerza klasę TraceTextWriter, implementuje interfejs ITraceXmlWriter, umożliwia tworzenie struktury Xml,
* MessageCounting Monitor – implementuje interfejs ITraceMonitor poprzez zliczanie komunikatów wysyłanych z określonej metody, Tworzy implementację klasy TraceTextWriter w metodzie Flush.

Takie rozwiązanie umożliwia powiązanie innych komponentów z modułem TraceWriting przez interfejsy i ewentualne zaimplementowanie tych interfejsów w innych komponentach.

## FileComparing

Moduł TestHelper zawiera cztery klasy przeznaczone do porównywania plików. Są to:

* FileCompareOptions – reprezentuje opcje porównywania. Umożliwia przycinanie spacji na końcach linii, traktowanie wielokrotnych spacji jak pojedynczych, ignorowanie wielkości liter itp.
* AbstractFileComparer – abstrakcyjna klasa do porównywania plików. Zawiera podstawowe operacje porównywania i wyświetlania linii. Podstawowa metoda CompareFiles musi być zaimplementowana w klasie potomnej. Wykorzystuje interfejs ITraceStreamWriter do wyświetlania niezgodnych linii.
* TextFileComparer – konkretna klasa porównująca pliki tekstowe. Realizuje szereg opcji porównania, w tym ignorowanie linii pustych. Gdy wykryje niezgodne ze sobą obszary linii, to próbuje zsynchronizować porównanie dalszych obszarów.
* XmlFileComparer – konkretna klasa porównująca pliki Xml. W odróżnieniu od porównywania tekstowego interpretuje wewnętrzną strukturę plików Xml.

## TestBuilding

Moduł TestHelper zawiera podstawową infrastrukturę do tworzenia testów. Są to klasy:

* AbstractTest – abstrakcyjna klasa bazowa wszystkich testów definiująca metodę Run, wirtualną metodę Init oraz abstrakcyjne metody Prepare, Execute i Finalize,
* AbstractFileGenTest – abstrakcyjna klasa bazowa dla testów polegających na generowaniu plików, zawiera porównywanie wyników.
* TestCase – abstrakcyjna klasa reprezentująca przypadek testowy, ma abstrakcyjną metodę Execute,
* TestCase<DataType> – klasa reprezentująca przypadek testowy przechowujący dane i wykonujący funkcję testu na tych danych.