

Laboratorium 11

Testy akceptacyjne z wykorzystaniem narzędzia FitNesse

Cel laboratorium:**Nabywanie umiejętności przygotowywania testów akceptacyjnych za pomocą narzędzia FitNesse**

1. Wg wskazówek podanych w **Dodatku 1 p. 1**, należy zainstalować narzędzie **FitNesse** oraz wykonać projekt w środowisku **NetBeans 8.2**. Następnie, wg kolejnych przykładów w **Dodatku 1**, należy dodawać podane dalej **testy akceptacyjne** wybranych funkcji oprogramowania zaprojektowanego podczas lab. 2-11, uruchamianych z pośrednictwem klasy typu **Facade** warstwy biznesowej. Poniżej, w kolejnych punktach instrukcji podano, ile i jakie testy należy wykonać w jedno- i dwu-osobowych grupach.
2. Należy wykonać testy akceptacyjne metod klasy typu **Facade**, która przetwarza dane, które będą wykorzystywane w testach kolejnych metod klasy typu **Facade**, zaproponowanych w kolejnych punktach.

W tabelce podano informację dotyczącą wyboru metod do testowania oraz przykładów rozwiązań.

Grupa	Liczba metod do testowania klasy opartej na wzorcu Facade	Przykłady metod wybranych do testowania	Przykłady testów
		Facade	Klasy testujące
1 osoba	1	addClient	Test_addClient (p.3.10, Dodatek1)
2 osoby	2	addClient, addTitleBook	Test_addClient (p.3.10, Dodatek1) Test_addTitleBook (p.3.11, Dodatek 1)

W p.3.6 **Dodatku 1** zdefiniowano tabelkę z wartościami wzorcowymi danych wejściowych i wyjściowych, wykorzystanych w testach akceptacyjnych klasy typu **Facade**. Należy opracować podobny zestaw danych wzorcowych, który należy wykorzystać przy budowie własnych testów akceptacyjnych.

3. Należy wykonać testy akceptacyjne metod klasy opartej na wzorcu **Facade**, które korzystają z wyników przetwarzania danych realizowanych przez testy z p. 2.

Grupa	Liczba metod do testowania klasy opartej na wzorcu Facade	Przykłady wybranych do testowania	Przykłady testów
		Facade	Klasy testujące
1 osoba	1	addTitleBook, addBook	Test_addTitleBook (p.3.11, Dodatek 1) Test_addBook(p.3.12, Dodatek 1)
2 osoby	2	addBook, addReservation	Test_addBook (p.3.12, Dodatek 1) Test_addReservation (p.3.13, Dodatek 1)

Należy rozszerzyć zestaw danych wzorcowych z p.2, który należy wykorzystać przy budowie własnych testów akceptacyjnych w p.3.

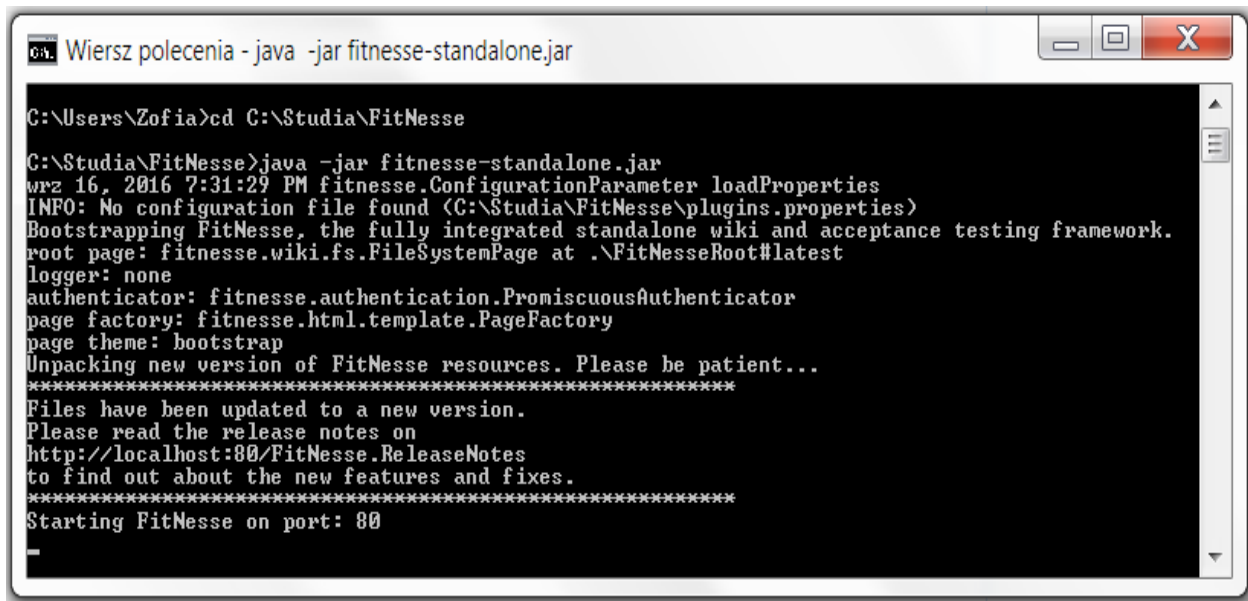
Dodatek 1

1. Instalacja narzędzia FitNesse - wersja v20160618

- 1.1. Należy pobrać oprogramowanie *FitNesse* ze strony <http://www.fitnessse.org>, które zawiera zintegrowane narzędzie do testowania oraz witrynę typu *Wiki* do tworzenia stron służących do uruchamiania oprogramowania testującego oprogramowanie.
- 1.2. Należy wykonać folder np. *FitNesse*, gdzie należy umieścić pobrany plik *fitnessse-standalone.jar* ze strony <http://www.fitnessse.org/FitNesseDownload>. Podczas tworzenia testów akceptacyjnych folder ten będzie służył również do przechowywania systemu stron służących do uruchamiania testów akceptacyjnych oprogramowania. W celu uruchomienia narzędzia należy z linii poleceń wpisać:

```
java -jar fitnessse-standalone.jar
```

Poniżej pokazano komunikaty po pierwszym uruchomieniu narzędzia.

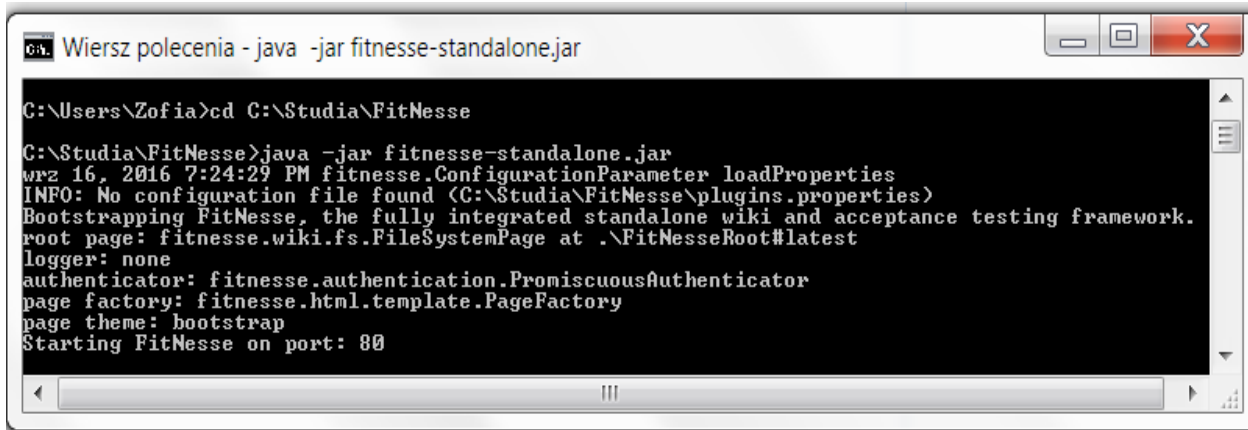


```
Wiersz polecenia - java -jar fitnessse-standalone.jar

C:\Users\Zofia>cd C:\Studia\FitNesse

C:\Studia\FitNesse>java -jar fitnessse-standalone.jar
wrz 16, 2016 7:31:29 PM fitnessse.ConfigurationParameter loadProperties
INFO: No configuration file found (C:\Studia\FitNesse\plugins.properties)
Bootstrapping FitNesse, the fully integrated standalone wiki and acceptance testing framework.
root page: fitnessse.wiki.fs.FileSystemPage at .\FitNesseRoot#latest
logger: none
authenticator: fitnessse.authentication.PromiscuousAuthenticator
page factory: fitnessse.html.template.PageFactory
page theme: bootstrap
Unpacking new version of FitNesse resources. Please be patient...
*****
Files have been updated to a new version.
Please read the release notes on
http://localhost:80/FitNesse.ReleaseNotes
to find out about the new features and fixes.
*****
Starting FitNesse on port: 80
_
```

Poniżej pokazano komunikaty po kolejnym uruchomieniu narzędzia *FitNesse*.

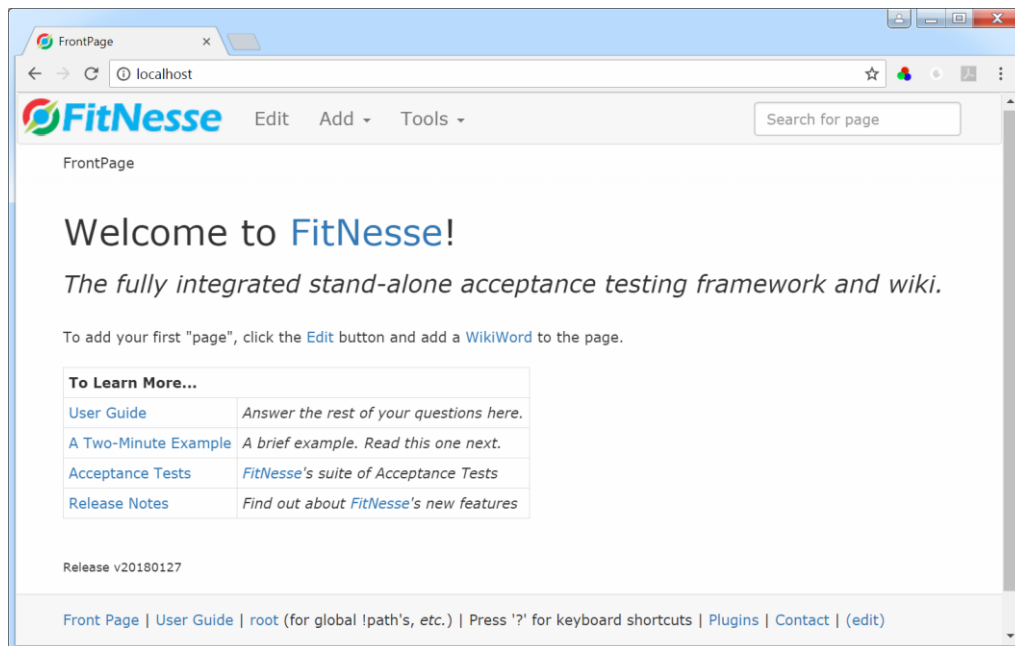


```
Wiersz polecenia - java -jar fitnessse-standalone.jar

C:\Users\Zofia>cd C:\Studia\FitNesse

C:\Studia\FitNesse>java -jar fitnessse-standalone.jar
wrz 16, 2016 7:24:29 PM fitnessse.ConfigurationParameter loadProperties
INFO: No configuration file found (C:\Studia\FitNesse\plugins.properties)
Bootstrapping FitNesse, the fully integrated standalone wiki and acceptance testing framework.
root page: fitnessse.wiki.fs.FileSystemPage at .\FitNesseRoot#latest
logger: none
authenticator: fitnessse.authentication.PromiscuousAuthenticator
page factory: fitnessse.html.template.PageFactory
page theme: bootstrap
Starting FitNesse on port: 80
```

Uruchomienie witryny typu **Wiki** nastąpi po wpisaniu w przeglądarce adresu <http://localhost>. Poniżej pokazano stronę główną uruchomionego środowiska do tworzenia i wykonania testów akceptacyjnych.



1.3. Jeśli nie startuje prawidłowo witryna typu **Wiki** narzędzia **FitNesse**, należy uruchomić to narzędzie w następujący sposób, podając **wolny numer portu** np.:

```
java -jar fitnessse-standalone.jar -p 8080
```

i w przeglądarce należy wtedy wpisać: <http://localhost:8080>.

1.4. Na stronie <http://localhost/FitNesse.UserGuide.WritingAcceptanceTests> jest dostępna informacja typu **Przewodnik użytkownika** w zakresie technologii testowania za pomocą **FitNesse**.

2. **Modyfikacje kodu przedstawionego w instrukcji do lab5-7 – identyczne zmiany należało wykonać podczas tworzenia testów jednostkowych.**

Dodanie generowania wyjątku w przypadku niepoprawnej wartości pierwszego elementu tablicy *data* -zmiana definicji podanej w instrukcji 6. Pełna walidacja poprawności danych wejściowych powinna być realizowana przez warstwę klienta aplikacji!

```
package subbusinessstier;
```

```
import java.time.LocalDate;
import java.util.IllegalFormatException;
import subbusinessstier.entities.Book;
import subbusinessstier.entities.Client;
import subbusinessstier.entities.Reservation;
import subbusinessstier.entities.TitleBook;
import subbusinessstier.entities.TitleBookRead;
```

```
public class Factory {
```

```
    public TitleBook createTitleBook(String data[]) {
        TitleBook titleBook = null;
        switch (Integer.parseInt(data[0])) //what_title_book_type
        {
            case 0:
                titleBook = new TitleBook(); //TTitle_book object for searching
                titleBook.setISBN(data[1]);
                break;
            case 1:
                titleBook = new TitleBook(); //TTitle_book object for persisting
                titleBook.setAuthor(data[1]);
                titleBook.setTitle(data[2]);
                titleBook.setISBN(data[3]);
                titleBook.setPublisher(data[4]);
                break;
            case 2:
                TitleBookRead title_book1 = new TitleBookRead(); //TTitle_book_on_tape object for searching
                title_book1.setISBN(data[1]);
                title_book1.setActor(data[2]);
                titleBook = title_book1;
                break;
            case 3:
                TitleBookRead title_book2 = new TitleBookRead(); //TTitle_book_on_tape object for persisting
                title_book2.setAuthor(data[1]);
                title_book2.setTitle(data[2]);
                title_book2.setISBN(data[3]);
                title_book2.setPublisher(data[4]);
                title_book2.setActor(data[5]);
                titleBook = title_book2;
                break;
            default:
                throw new IllegalFormatException(0); //generowanie wyjątku z powodu niepoprawnej
                //wartości elementu tablicy dane o indeksie 0.
        }
        return titleBook;
    }
}
```

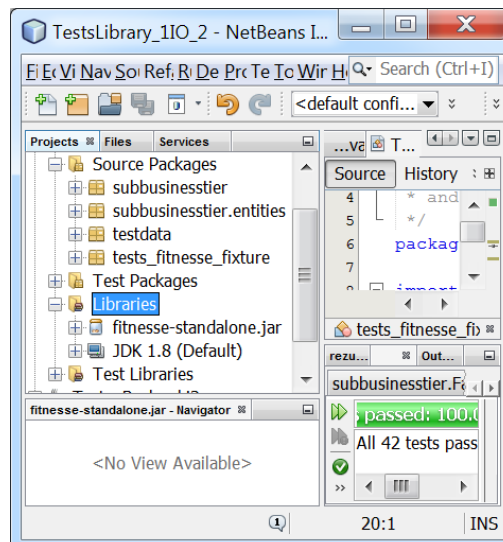
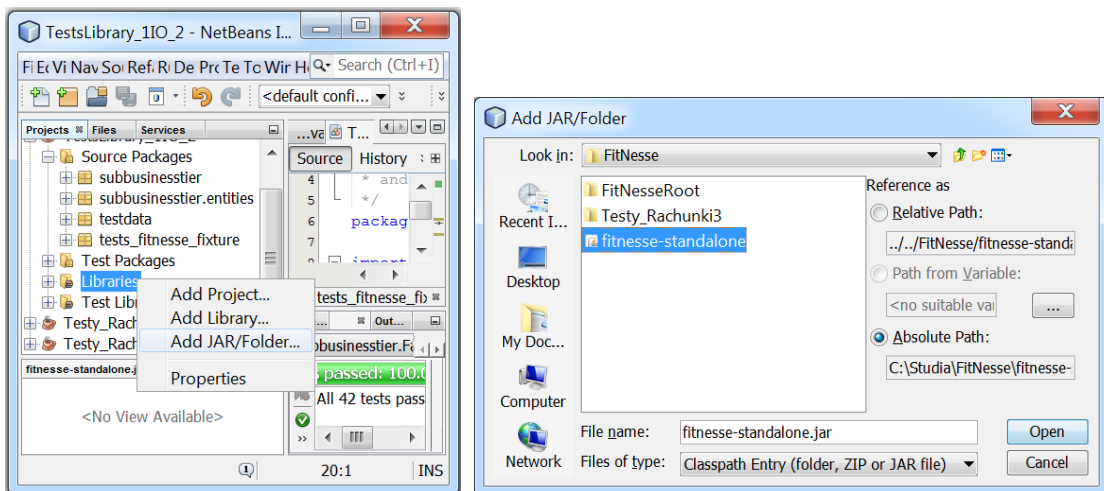
```
public Book createBook(String data[]) {
    Book book = null;
    switch (Integer.parseInt(data[0])) //what_book_type
    {
        case 0:
            book = new Book();//TBook object for persisting
            book.setNumber(Integer.parseInt(data[1]));
            break; }
    return book;
}

public Client createClient(String data[]) {
    Client client = null;
    switch (Integer.parseInt(data[0])) //what_book_type
    {
        case 0:
            client = new Client();
            client.setNumberCard(Integer.parseInt(data[1]));
            break;
        case 1:
            client = new Client();
            client.setName(data[1]);
            client.setNumberCard(Integer.parseInt(data[2]));
            break; }
    return client;
}

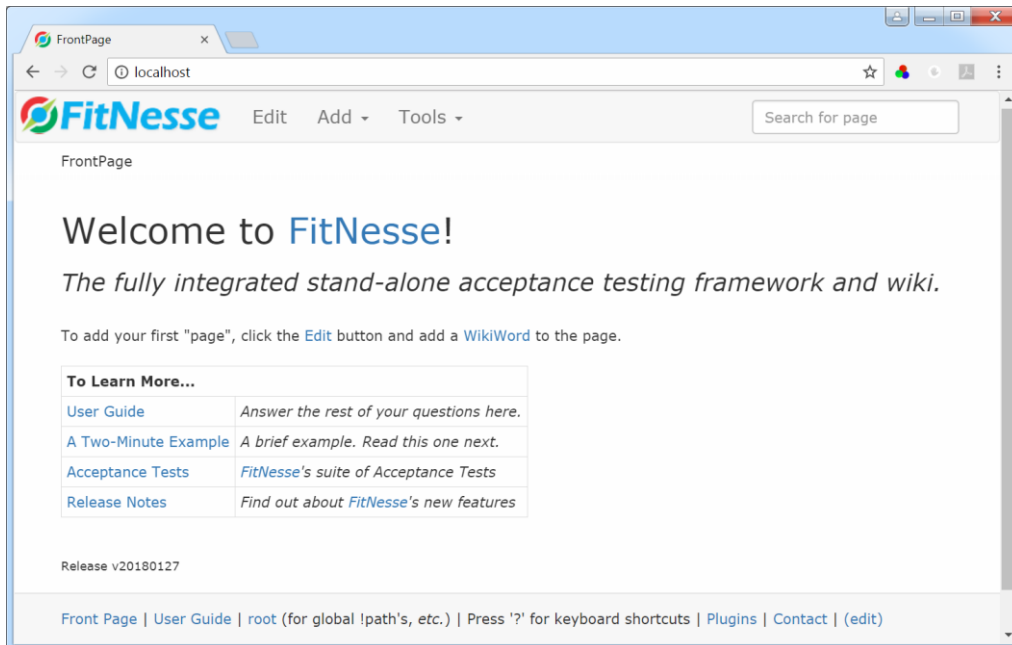
public Reservation createReservation(int number, LocalDate date) {
    Reservation reservation = new Reservation();
    reservation.setNumber(number);
    reservation.setDate(date);
    return reservation;
}
}
```

3. Przykład tworzenia testów akceptacyjnych warstwy biznesowej aplikacji

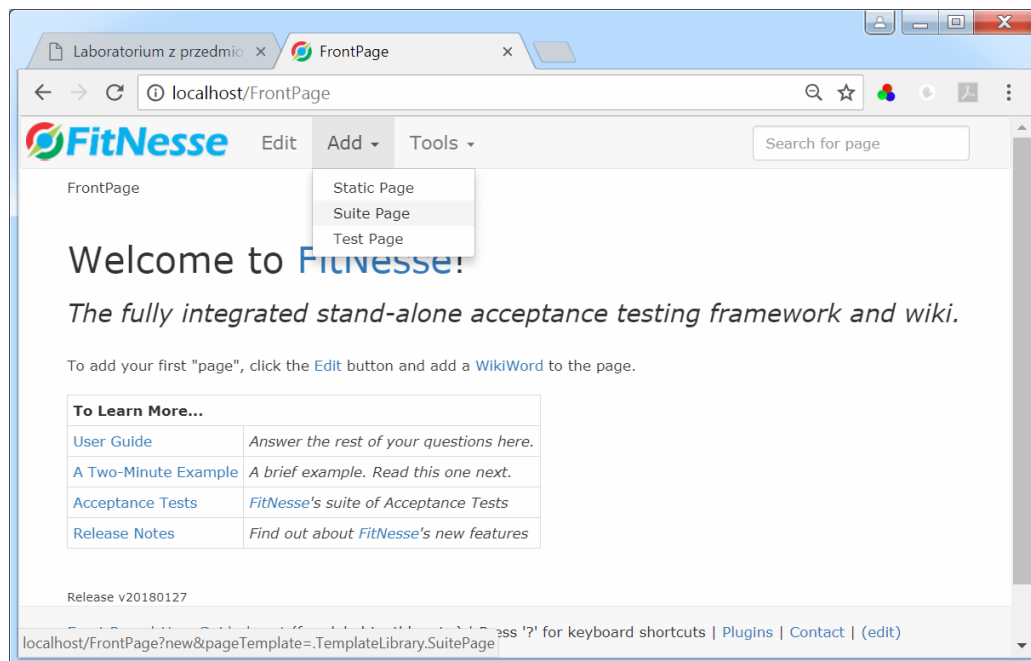
- 3.1. W celu utworzenia projektu zawierającego kod do testowania, należy wybrać w **Menu Bar** pozycję **Files**. Na tej liście kliknąć na pozycję **New Project**. W oknie **New Project**, w liście **Categories** należy wybrać pozycję **Java**, a w liście **Projects** należy wybrać pozycję **Java Class Library** i kliknąć na przycisk **Next**. W kolejnym formularzu należy wpisać nazwę projektu w polu **Project Name** i wybrać położenie projektu w polu **Project Location**. W przykładzie projekt ma nazwę **TestsLibrary_1IO_2**.
- 3.2. W zakładce **Projects**, w folderze **Source Packages** umieścić kopię pakietu z oprogramowaniem do testowania, wykonanym podczas lab 2- lab 11. W przykładzie są to pakiety **subbusinessstier** oraz **subbusinessstier.entities**, wykorzystany podczas testów jednostkowych. Dodatkowo, należy utworzyć pakiet o nazwie **testdata**.
- 3.3. W zakładce **Projects**, w folderze **Source Packages** należy wykonać nowy pakiet, w którym umieszczane będą klasy realizujące testy akceptacyjne - w przykładzie jest to pakiet **tests_fitness_fixture**.
- 3.4. W zakładce **Libraries** utworzonego projektu należy dodać plik **fitnessse-standalone.jar** w następujący sposób:



- 3.5. Należy wykonać stronę internetową, która będzie przechowywać połączenia do poszczególnych stron internetowych reprezentujących testy akceptacyjne poszczególnych funkcji warstwy biznesowej.

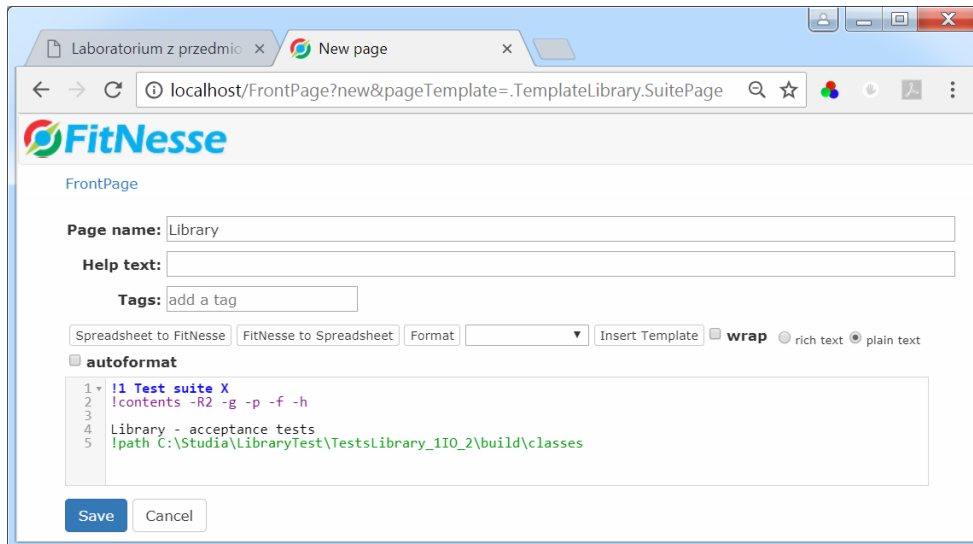


W tym celu należy dodać do adresu strony narzędzia **FitNesse**, uruchomionego w p. 1.2 lub 1.3, łańcuch **/FrontPage**. Następnie, w **Menu Bar** wybrać pozycję **Add** i następnie pozycję **Suite Page** (rysunek poniżej).

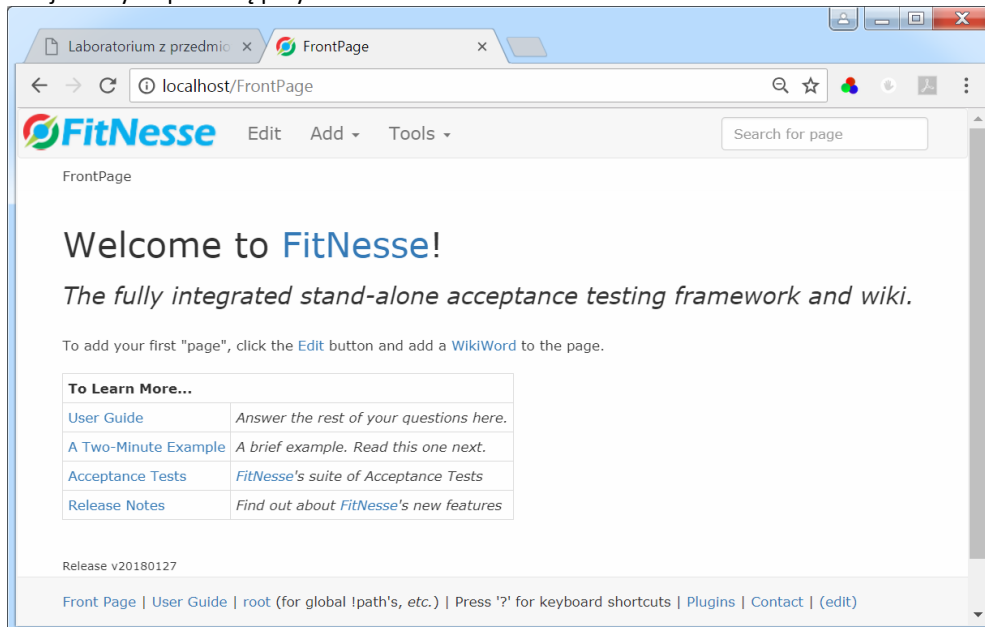


Dalej, w formularzu strony typu **Suite Page**, jako głównej strony testów akceptacyjnych, nadano nazwę **Library** w polu **Page name**. W celu tworzenia zestawu połączeń do kolejnych testów należy zachować znacznik **!contents** umożliwiający umieszczanie połączeń do kolejnych stron (tworzonych w dalszej części instrukcji) uruchamiających testy akceptacyjne. **Uwaga: Nazwy stron powinny w nazwie zawierać przynajmniej jedną dużą literę!!!**

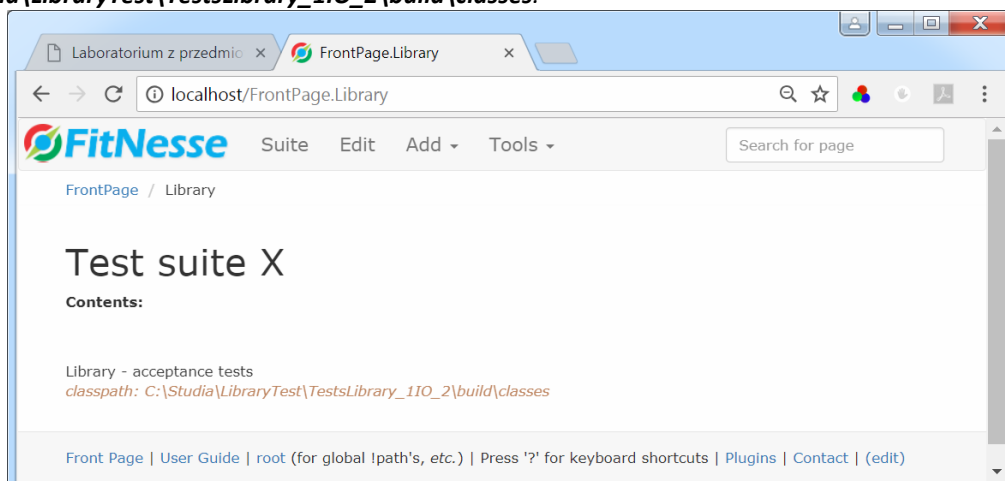
W zawartości strony umieszczono komentarz dotyczący przedmiotu testowania oraz ścieżkę dostępu do kodu obsługującego testowanie w katalogu*build\classes* projektu.



Poniżej przedstawiono widok strony głównej systemu testów akceptacyjnych **Library**, po zatwierdzeniu zawartości tej strony za pomocą przycisku **Save**.



Po dodaniu do adresu strony łańcuch: **.Library** widok strony podano poniżej. W przykładzie wykonano projekt **TestsLibrary_110_2** (p.3.1), zawierający kod do testowania w pakietach **subbusinessstier** oraz **subbusinessstier.entities** oraz kod testujący w pakiecie **tests_fitness_fixture**, dostępne w katalogu **C:\Studia\LibraryTest\TestsLibrary_110_2\build\classes**.



- 3.6. W projekcie utworzonym w p.3.1-3.3, w utworzonym pakiecie **testdata** należy utworzyć klasę **Data**, zawierającą dane wzorcowe do testowania.

```
package testdata;

public class Data1 {
public String databooks1[]={"\nTitle: Title1 Author: Author1 ISBN: ISBN1 Publisher: Publisher1 Number: 1",
"\nTitle: Title1 Author: Author1 ISBN: ISBN1 Publisher: Publisher1 Number: 2"};
public String databooks2[]={
"\nTitle: Title4 Author: Author4 ISBN: ISBN4 Publisher: Publisher4 Actor: Actor4 Number: 1",
"\nTitle: Title4 Author: Author4 ISBN: ISBN4 Publisher: Publisher4 Actor: Actor4 Number: 2"};
public String databooks3[]={"\nTitle: Title1 Author: Author1 ISBN: ISBN2 Publisher: Publisher1 Number: 1",
"\nTitle: Title1 Author: Author1 ISBN: ISBN1 Publisher: Publisher1 Number: 2"};
public String databooks[]={databooks1,databooks2,databooks3};
public String dataclients[]={ "\nClient{ numberCard=1, name=Klient1\nreservations=[]}",
"\nClient{ numberCard=2, name=Klient2\nreservations=[]}"};
public String datatitles[]={"\nTitle: Title1 Author: Author1 ISBN: ISBN1 Publisher: Publisher1",
"\nTitle: Title2 Author: Author2 ISBN: ISBN2 Publisher: Publisher2",
"\nTitle: Title3 Author: Author3 ISBN: ISBN3 Publisher: Publisher3",
"\nTitle: Title1 Author: Author1 ISBN: ISBN1 Publisher: Publisher1 Actor: Actor1",
"\nTitle: Title2 Author: Author2 ISBN: ISBN2 Publisher: Publisher2 Actor: Actor2",
"\nTitle: Title4 Author: Author4 ISBN: ISBN4 Publisher: Publisher4 Actor: Actor4"};
public String reservationclients[]={
"\nClient{ numberCard=1, name=Klient1+\nreservations=[Reservation{ number=0, date=2018-01-20}]",
"\nClient{ numberCard=1, name=Klient1+\nreservations=[Reservation{ number=0, date=2018-01-20},
Reservation{ number=1, date=2018-01-20}]",
"\nClient{ numberCard=2, name=Klient2+\nreservations=[Reservation{ number=2, date=2018-01-20}]}"};
}
}
```

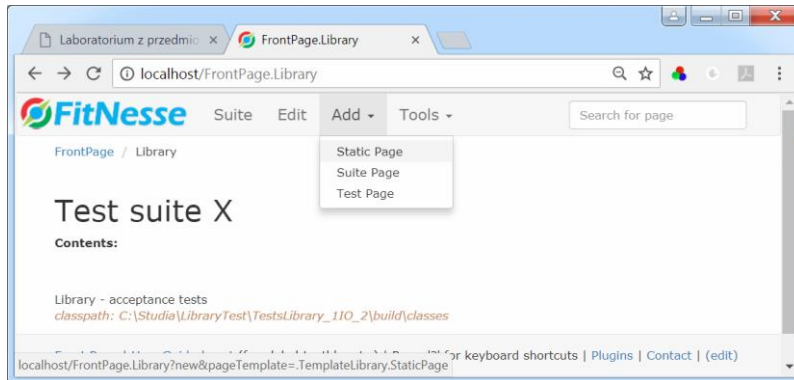
- 3.7. W projekcie utworzonym w p.3.1-3.3, w utworzonym pakiecie **tests_fitness_fixture** należy utworzyć klasę o nazwie **SetUp**, która będzie inicjowała obiekty testowane w testach akceptacyjnych. Poniżej podano kod klasy **SetUp** z przykładu:

```
package tests_fitness_fixture;

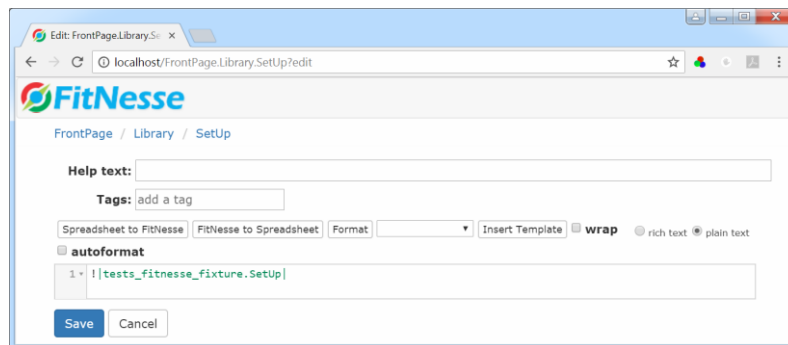
import fit.Fixture;
import subbusinessier.Facade;
import testdata.Data1;

public class SetUp extends Fixture{
    static Facade facade;
    static Data1 data;
    public SetUp()
    { facade = new Facade();
    data=new Data1(); }
}
```

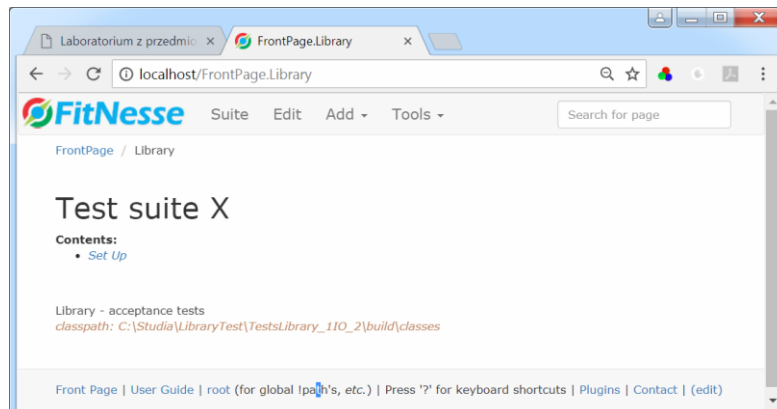
3.8. Należy wykonać stronę o nazwie **SetUp** typu **Static Page** - po przejściu na stronę główną **Library** należy w **Menu Bar** wybrać pozycję **Add** i następnie pozycję **Static Page** (rysunek poniżej).



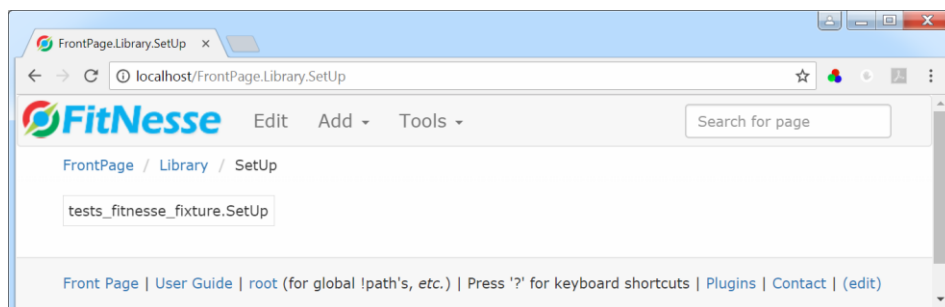
W polu **Page name** należy wpisać nazwę **SetUp** (podobnie jak przy tworzeniu strony głównej **Library**). Widok poniżej pokazuje stronę w stanie edycji zawartości strony – stan po zamknięciu strony **SetUp** po jej założeniu i ponowne otwarcie jej za pomocą opcji **Edit** z **Menu Bar**. Podczas edycji należy wpisać wartość względnej ścieżki pakietowej klasy **SetUp**. Strona typu **SetUp** wskazuje na kod wykonanej klasy **SetUp** w pakiecie **tests_fitness_fixture**. Edycję strony należy zatwierdzić za pomocą przycisku **Save**.



Poniżej przedstawiono stronę główną **Library** typu **Suite Page** zawierającą połączenie do strony **SetUp** typu **Static Page**.

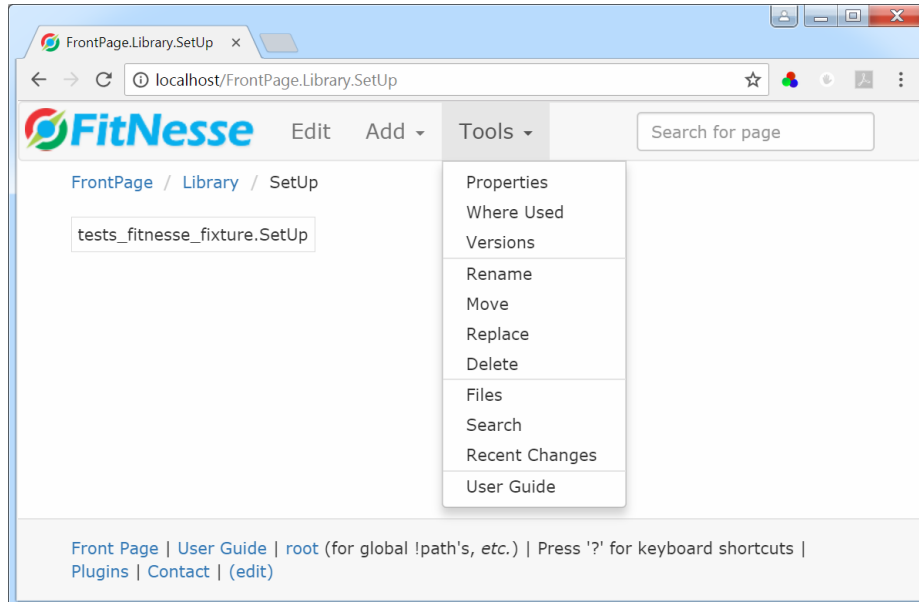


Poniżej przedstawiono widok strony **SetUp**.



- 3.9. Testy akceptacyjne klasy *Facade* opierają się na wywołaniu głównych testowanych metod *addClient*, *addTitleBook*, *addBook*, *addReservation* oraz pomocniczych metod w metodach klas testujących, dziedziczących po klasie *ColumnFixture*. Opis tworzenia testów akceptacyjnych w środowisku uruchomionego narzędzia FitNesse w 1.2 lub 1.3 jest dostępny z *MenuBar/Tools/User Guide* (rysunek poniżej), czyli:

<http://localhost/FitNesse.UserGuide.WritingAcceptanceTests>.



Przed wywołaniem każdej metody testującej lub grup metod testujących tworzony jest obiekt typu *SetUp*, który tworzy obiekt typu *Facade*, oparty na koncepcji klasy typu *Facade* warstwy biznesowej testowanej aplikacji. Dalej przedstawiono definicję klas testujących poszczególne funkcje oprogramowania oraz stron uruchamiających te testy.

- 3.10. Jako pierwszy, należy dodać test akceptacyjny dodawania klienta, czyli metody *addClient* klasy *Facade*. W projekcie utworzonym w p.3.1-3.3 należy dodać nową klasę *Test_addClient* do pakietu *tests_fitness_fixture*. Test ten sprawdza wynik metody *addClient* klasy *Facade* podczas podawania danych bez naruszenia integralności danych i z naruszeniem integralności obiektów typu *Client* testując wynikiem porównania danych zwracanych przez metodę (zmienna *result*) i wynik spodziewany (*data*). Kolorem czerwonym zaznaczono nazwy metod i atrybutów, zastosowanych dalej przy budowie tablicy decyzyjnej testu na stronie *addClient*.

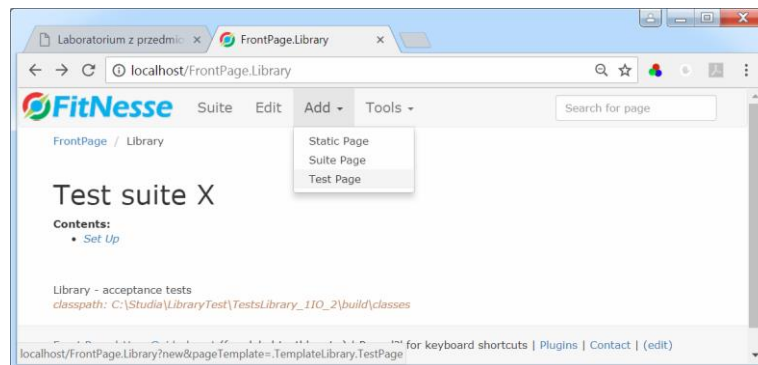
```
package tests_fitness_fixture;

import fit.ColumnFixture;

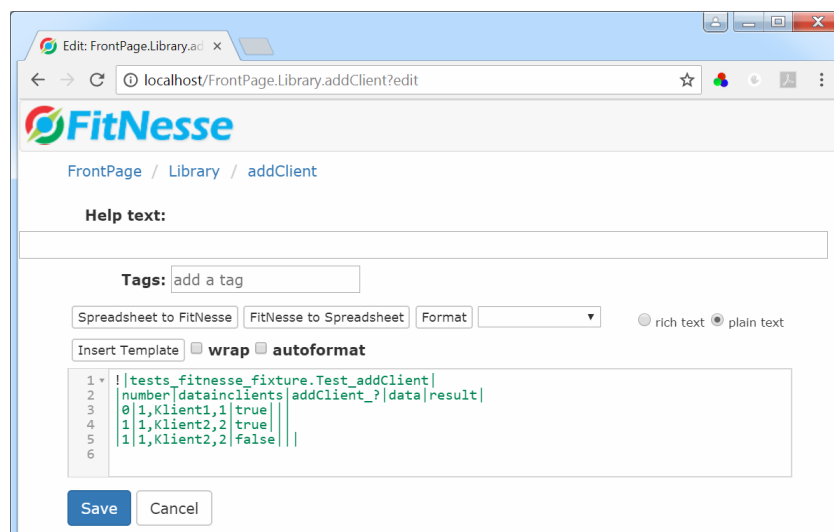
public class Test_addClient extends ColumnFixture{
    String datainclients[], data, result;
    int number;

    public boolean addClient_()
    { result=null;
      result=SetUp.facade.addClient(datainclients);
      data=SetUp.data.dataclients[number];
      return data.equals(result);
    }
}
```

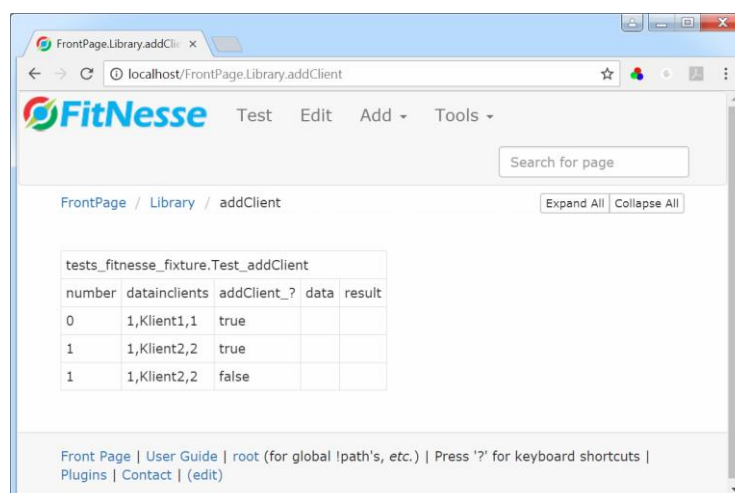
Należy dodać stronę **addClient_** z testem akceptacyjnym dodawania klienta przez wybór na stronie **Library** z listy **Menu Bar/Add** pozycji **Test Page** (rysunek poniżej). Strona ta umożliwi uruchomienie testu akceptacyjnego realizowanego przez klasę **Test_addClient**, dodaną do pakietu **tests_fitness_fixture**. Wykonanie strony **addClient** uruchamiającej test akceptacyjny realizowany przez klasę **Test_addClient** pokazano na kolejnych rysunkach p.3.10. Testy wykonano w oparciu o dane wzorcowe z klasy **Data** (p.3.6).



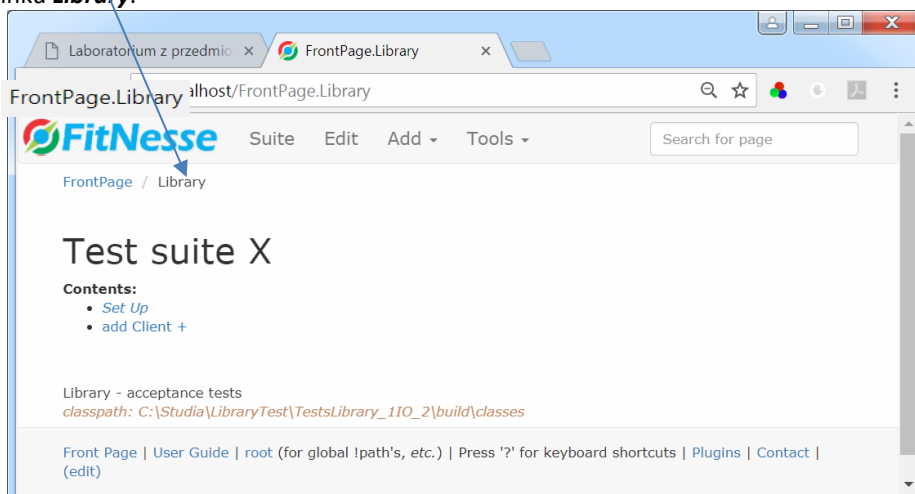
Utworzenie tabeli testującej (tabeli decyzyjnej) - wyniki zwracane przez metodę **addClient** po przekazaniu danych w tablicy **datainclients** przedstawiono na rysunku poniżej (stan po utworzeniu strony, podobnie jak strony **SetUP** i zamkniętej po naciśnięciu klawisza **Save**, a następnie otwartej za pomocą opcji **Edit**, wybranej z **Menu Bar**) – kolumny **datainclients** oraz **number** zawierają dane testowe, przekazane do metody testującej. Kolumna **addClient_?** zawiera dane porównywane z danymi zwracanymi przez metodę testującą **addClient_**. Kolumna **data** wyświetla dane wzorcowe pobrane z tabeli **dataclients** z klasy **Data** wyznaczone za pomocą podanego indeksu **number**, a kolumna **result** podaje wyniki zwracane przez metodę testowaną **addClient**.



Dalej przedstawiono widok strony **addClient** po zatwierdzeniu jej zawartości po naciśnięciu przycisku **Save**.



Poniżej przedstawiono widok strony głównej **Library** po zatwierdzeniu zawartości strony **addClient** i po naciśnięciu linku **Library**.



Widok strony testowej po uruchomieniu testu dodawania klienta za pomocą pozycji **Test** z **Menu Bar** strony **addClient** przedstawiono poniżej.

number	datainclients	addClient_?	data	result
0	1,Klient1,1	true	Client{ numberCard=1, name=Klient1 reservations=[]}	Client{ numberCard=1, name=Klient1 reservations=[]}
1	1,Klient2,2	true	Client{ numberCard=2, name=Klient2 reservations=[]}	Client{ numberCard=2, name=Klient2 reservations=[]}
1	1,Klient2,2	false	Client{ numberCard=2, name=Klient2 reservations=[]}	null

3.11. Wykonanie testu akceptacyjnego dodawania obiektów z rodziny *TitleBook* przez klasę *Facade* oraz kontrolę poprawności danych przekazanych do klasy *Factory*, przechwytyjąc wyjątek generowany przez klasę *Factory* w przypadku niepoprawnych danych. Kolorem czerwonym zaznaczono nazwy metod i atrybutów, zastosowanych dalej przy budowie tablicy decyzyjnej testu na stronie *addTitleBook*.

```
package tests_fitnessse_fixture;

import fit.ColumnFixture;
import java.util.IllegalFormatCodePointException;

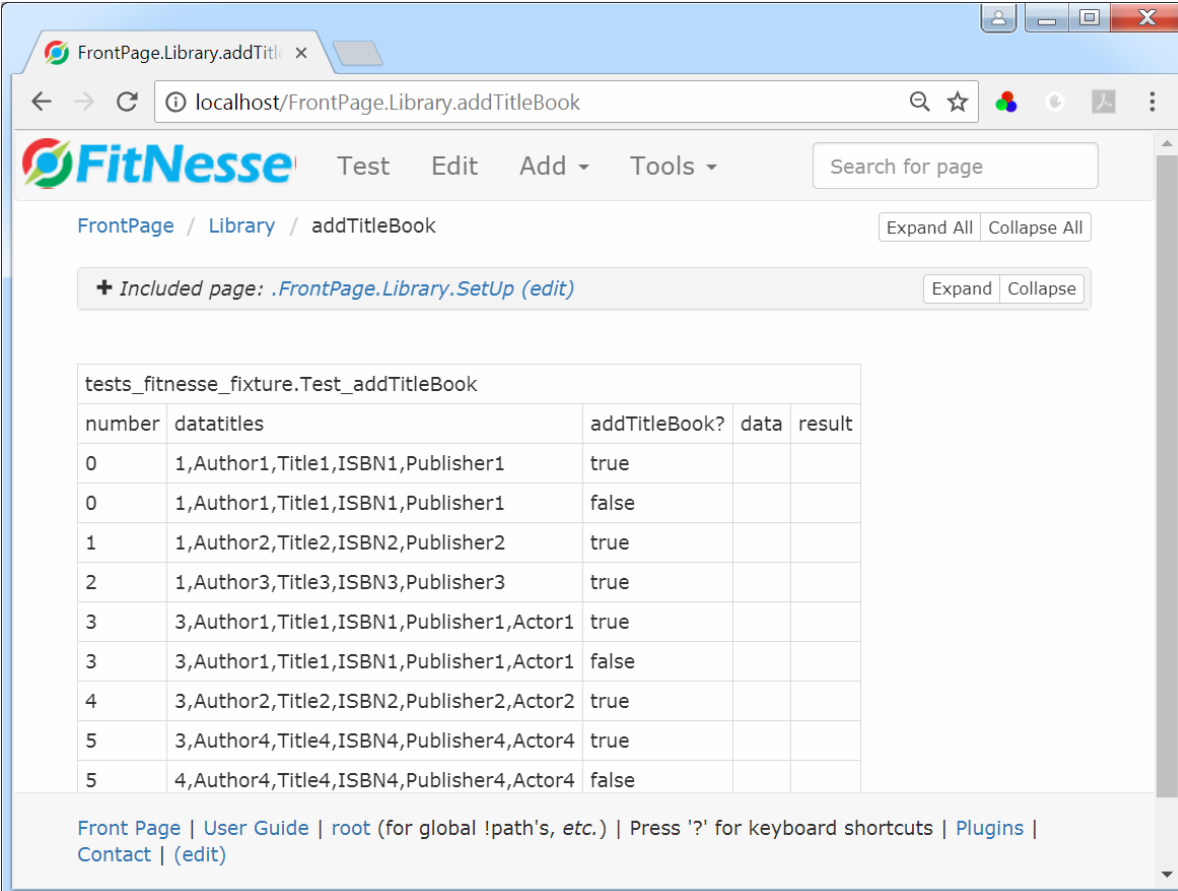
public class Test_addTitleBook extends ColumnFixture{
    String datatitles[],data,result;
    int number;

    public boolean addTitleBook() throws IllegalFormatCodePointException
    {
        try{
            result=null;
            result=SetUp.facade.addTitleBook(datatitles);
            data=SetUp.data.datatitles[number];
        } catch(IllegalFormatCodePointException e)
        {
            return false;
        }
        return data.equals(result);
    }
}
```

Wykonanie strony *addTitleBook* (podobnie jak strony *addClient*) i tablicy decyzyjnej do testowania dodawanie obiektów z rodziny *TitleBook*:

number	datatitles	addTitleBook?	data	result
0	1, Author1, Title1, ISBN1, Publisher1	true		
0	1, Author1, Title1, ISBN1, Publisher1	false		
1	1, Author2, Title2, ISBN2, Publisher2	true		
2	1, Author3, Title3, ISBN3, Publisher3	true		
3	3, Author1, Title1, ISBN1, Publisher1, Actor1	true		
3	3, Author1, Title1, ISBN1, Publisher1, Actor1	false		
4	3, Author2, Title2, ISBN2, Publisher2, Actor2	true		
5	3, Author4, Title4, ISBN4, Publisher4, Actor4	true		
5	4, Author4, Title4, ISBN4, Publisher4, Actor4	false		

Widok strony **addTitleBook** po naciśnięciu przycisku **Save**:



FrontPage.Library.addTitl

localhost/FrontPage.Library.addTitleBook

FitNesse Test Edit Add Tools Search for page

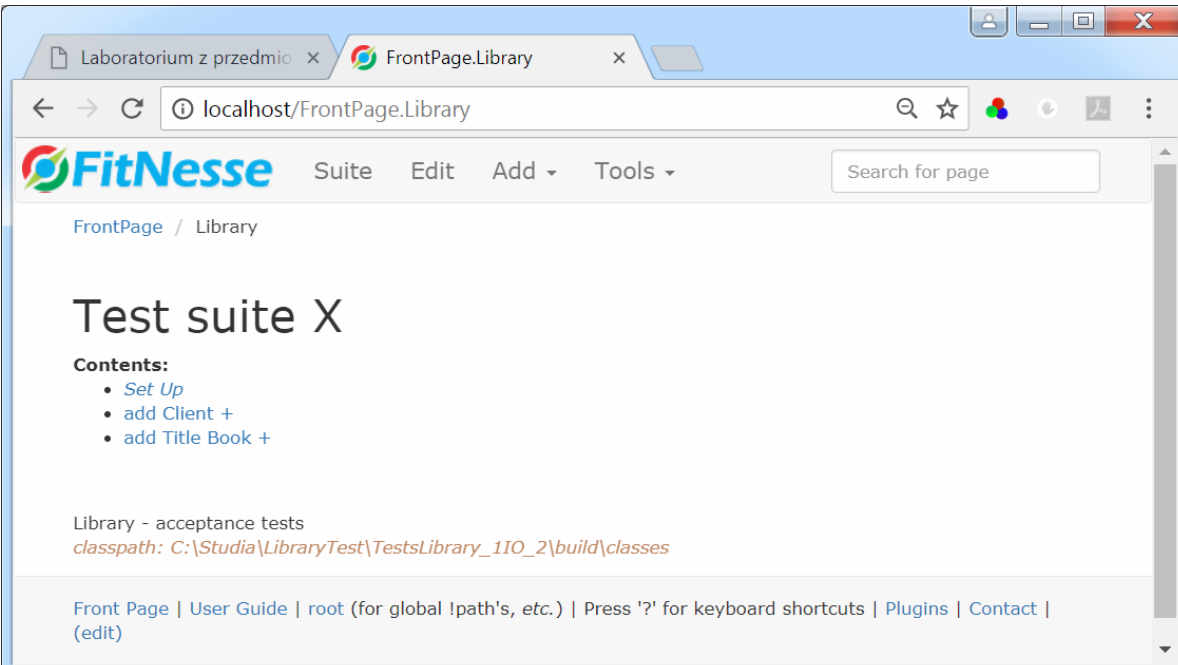
FrontPage / Library / addTitleBook Expand All Collapse All

+ Included page: [.FrontPage.Library.SetUp \(edit\)](#) Expand Collapse

tests_fitnesses_fixture.Test_addTitleBook				
number	datatitles	addTitleBook?	data	result
0	1,Author1,Title1,ISBN1,Publisher1	true		
0	1,Author1,Title1,ISBN1,Publisher1	false		
1	1,Author2,Title2,ISBN2,Publisher2	true		
2	1,Author3,Title3,ISBN3,Publisher3	true		
3	3,Author1,Title1,ISBN1,Publisher1,Actor1	true		
3	3,Author1,Title1,ISBN1,Publisher1,Actor1	false		
4	3,Author2,Title2,ISBN2,Publisher2,Actor2	true		
5	3,Author4,Title4,ISBN4,Publisher4,Actor4	true		
5	4,Author4,Title4,ISBN4,Publisher4,Actor4	false		

Front Page | User Guide | root (for global !path's, etc.) | Press '?' for keyboard shortcuts | Plugins | Contact | (edit)

Widok strony głównej **Library**:



Laboratorium z przedmio x FrontPage.Library x

localhost/FrontPage.Library

FitNesse Suite Edit Add Tools Search for page

FrontPage / Library

Test suite X

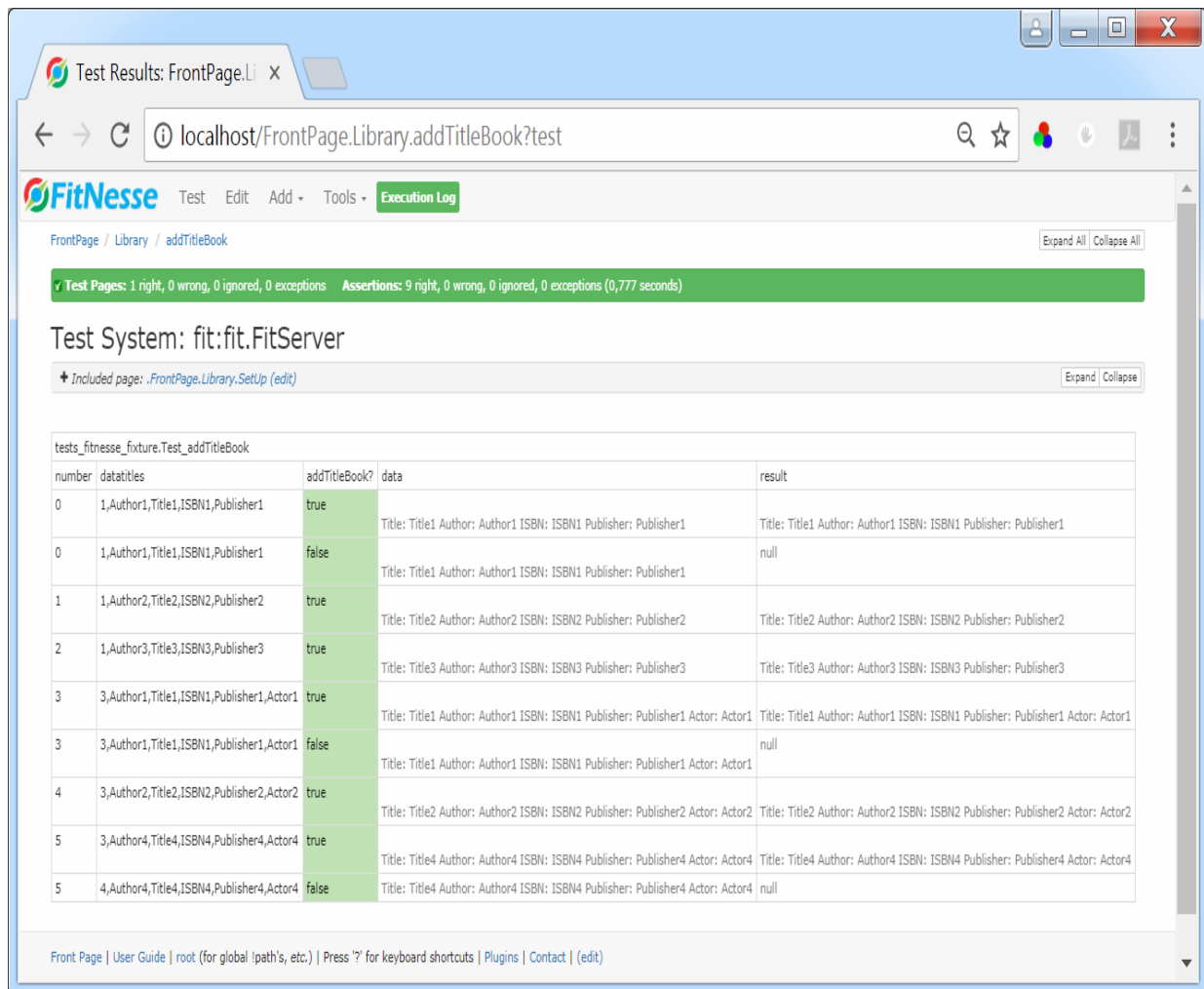
Contents:

- [Set Up](#)
- [add Client +](#)
- [add Title Book +](#)

Library - acceptance tests
classpath: C:\Studia\LibraryTest\TestsLibrary_1IO_2\build\classes

Front Page | User Guide | root (for global !path's, etc.) | Press '?' for keyboard shortcuts | Plugins | Contact | (edit)

Widok strony **addTitleBook** po przeprowadzeniu testów po naciśnięciu pozycji **Test** w **MenuBar** strony:



Test Results: FrontPage.LI x

localhost/FrontPage.Library.addTitleBook?test

FitNesse Test Edit Add Tools Execution Log

FrontPage / Library / addTitleBook Expand All Collapse All

✓ Test Pages: 1 right, 0 wrong, 0 ignored, 0 exceptions Assertions: 9 right, 0 wrong, 0 ignored, 0 exceptions (0,777 seconds)

Test System: fit:fit.FitServer

+ Included page: .FrontPage.Library.SetUp (edit) Expand Collapse

number	data:titles	addTitleBook?	data	result
0	1,Author1,Title1,ISBN1,Publisher1	true	Title: Title1 Author: Author1 ISBN: ISBN1 Publisher: Publisher1	Title: Title1 Author: Author1 ISBN: ISBN1 Publisher: Publisher1
0	1,Author1,Title1,ISBN1,Publisher1	false	Title: Title1 Author: Author1 ISBN: ISBN1 Publisher: Publisher1	null
1	1,Author2,Title2,ISBN2,Publisher2	true	Title: Title2 Author: Author2 ISBN: ISBN2 Publisher: Publisher2	Title: Title2 Author: Author2 ISBN: ISBN2 Publisher: Publisher2
2	1,Author3,Title3,ISBN3,Publisher3	true	Title: Title3 Author: Author3 ISBN: ISBN3 Publisher: Publisher3	Title: Title3 Author: Author3 ISBN: ISBN3 Publisher: Publisher3
3	3,Author1,Title1,ISBN1,Publisher1,Actor1	true	Title: Title1 Author: Author1 ISBN: ISBN1 Publisher: Publisher1 Actor: Actor1	Title: Title1 Author: Author1 ISBN: ISBN1 Publisher: Publisher1 Actor: Actor1
3	3,Author1,Title1,ISBN1,Publisher1,Actor1	false	Title: Title1 Author: Author1 ISBN: ISBN1 Publisher: Publisher1 Actor: Actor1	null
4	3,Author2,Title2,ISBN2,Publisher2,Actor2	true	Title: Title2 Author: Author2 ISBN: ISBN2 Publisher: Publisher2 Actor: Actor2	Title: Title2 Author: Author2 ISBN: ISBN2 Publisher: Publisher2 Actor: Actor2
5	3,Author4,Title4,ISBN4,Publisher4,Actor4	true	Title: Title4 Author: Author4 ISBN: ISBN4 Publisher: Publisher4 Actor: Actor4	Title: Title4 Author: Author4 ISBN: ISBN4 Publisher: Publisher4 Actor: Actor4
5	4,Author4,Title4,ISBN4,Publisher4,Actor4	false	Title: Title4 Author: Author4 ISBN: ISBN4 Publisher: Publisher4 Actor: Actor4	null

Front Page | User Guide | root (for global |path's, etc.) | Press '?' for keyboard shortcuts | Plugins | Contact | (edit)

3.12. Wykonanie testu akceptacyjnego dodawania książek przez testowaną aplikację – definicja klasy **Test_addBook** zawierającej kod do testowania tej funkcji, czyli metody **addBook** klasy **Facade**. **Kolorem czerwonym zaznaczono nazwy metod i atrybutów, zastosowanych przy budowie tablicy decyzyjnej testu na stronie **addBook**.**

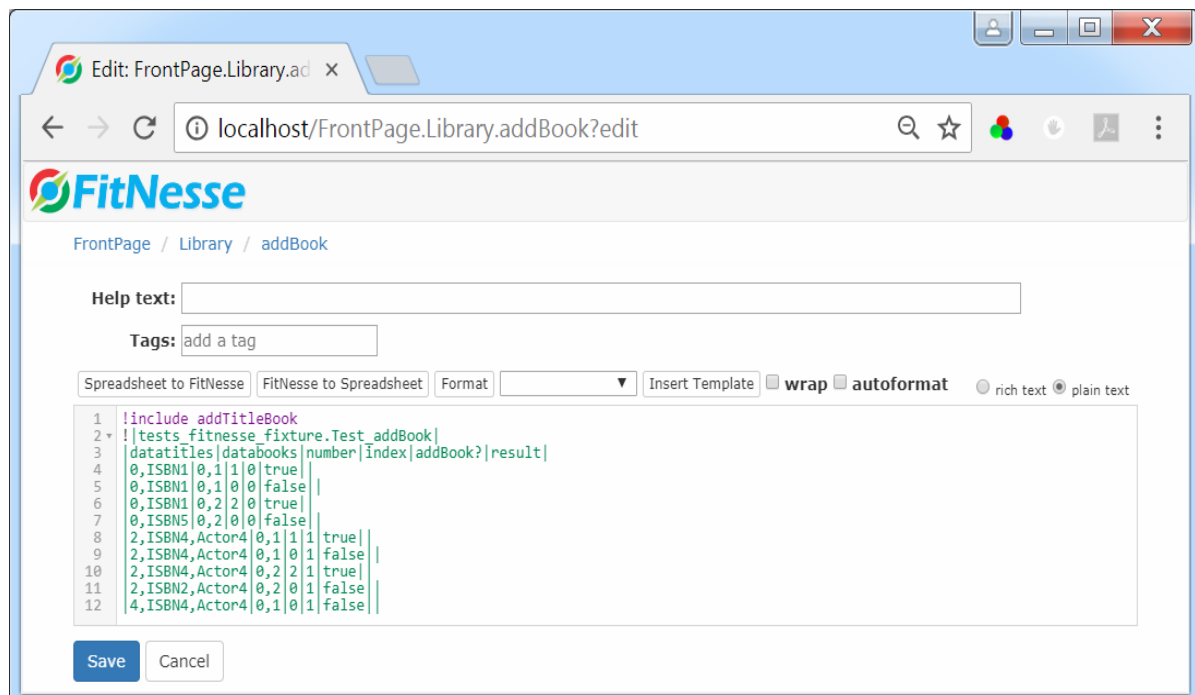
```
package tests_fitnessse_fixture;

import fit.ColumnFixture;
import java.util.ArrayList;
import java.util.IllegalFormatException;

public class Test_addBook extends ColumnFixture{
    String datatitles[], databooks[];
    int number, index;
    boolean result;

    public boolean addBook() {
        try {
            ArrayList<String> books=SetUp.facade.addBook(datatitles, databooks);
            if(books==null)
                return result=false;
            result= books.get(number-1).equals(SetUp.data.databooks[index][number-1]);
        } catch (IllegalFormatException e) {
            return false;
        }
        return result;
    }
}
```

Definicja strony **addBook** do testowania dodawania nowej książki (rysunek poniżej) opiera się na czynnościach takich samych, jak przy tworzeniu testu dodawania klienta w p.3.10. W treści strony należy uruchomić jeden poprzedni test dotyczący dodawania tytułu, aby na podstawie wprowadzonych danych podczas testowania wprowadzania tytułów książek wykonać testy akceptacyjne procesu dodawania książek. Dokonano tego dodając znacznik **!include** i podając nazwę strony testowej: **addTitleBook**. Dane wzorcowe testu pobrano z klasy **Data**.



Widok strony **addBook** po naciśnięciu przycisku **Save**:

The screenshot shows the FitNesse web browser interface. The browser address bar displays `localhost/FrontPage.Library.addBook`. The page title is `FrontPage / Library / addBook`. The page content includes:

- Included page: `.FrontPage.Library.SetUp (edit)`
- Included page: `addTitleBook (edit)`

The main content area displays two test tables:

tests_fitnesses_fixture.Test_addTitleBook

number	datatitles	addTitleBook?	data	result
0	1,Author1,Title1,ISBN1,Publisher1	true		
0	1,Author1,Title1,ISBN1,Publisher1	false		
1	1,Author2,Title2,ISBN2,Publisher2	true		
2	1,Author3,Title3,ISBN3,Publisher3	true		
3	3,Author1,Title1,ISBN1,Publisher1,Actor1	true		
3	3,Author1,Title1,ISBN1,Publisher1,Actor1	false		
4	3,Author2,Title2,ISBN2,Publisher2,Actor2	true		
5	3,Author4,Title4,ISBN4,Publisher4,Actor4	true		
5	4,Author4,Title4,ISBN4,Publisher4,Actor4	false		

tests_fitnesses_fixture.Test_addBook

datatitles	databooks	number	index	addBook?	result
0,ISBN1	0,1	1	0	true	
0,ISBN1	0,1	0	0	false	
0,ISBN1	0,2	2	0	true	
0,ISBN5	0,2	0	0	false	
2,ISBN4,Actor4	0,1	1	1	true	
2,ISBN4,Actor4	0,1	0	1	false	
2,ISBN4,Actor4	0,2	2	1	true	
2,ISBN2,Actor4	0,2	0	1	false	
4,ISBN4,Actor4	0,1	0	1	false	

The footer of the page contains navigation links: `Front Page | User Guide | root (for global !path's, etc.) | Press '?' for keyboard shortcuts | Plugins | Contact | (edit)`.

Poniżej podano widok strony głównej **Library** po dodaniu nowej strony testowej **addBook**.

The screenshot shows the FitNesse web browser interface. The browser address bar displays `localhost/FrontPage.Library`. The page title is `FrontPage / Library`. The page content includes:

- Library
- Contents:
 - [Set Up](#)
 - [add Book +](#)
 - [add Client +](#)
 - [add Title Book +](#)

Below the contents, there is a section for "Library - acceptance tests" with the classpath: `C:\Studia\LibraryTest\TestsLibrary_11O_2\build\classes`.

The footer of the page contains navigation links: `Front Page | User Guide | root (for global !path's, etc.) | Press '?' for keyboard shortcuts | Plugins | Contact | (edit)`.

Widok strony testowej po uruchomieniu testu dodawania nowej książki za pomocą pozycji **Test** z **Menu Bar** strony **addBook** pokazano na rysunku poniżej.

The screenshot shows the FitNesse test results for the 'addBook' test. The browser address bar shows 'localhost/FrontPage.Library.addBook?test'. The FitNesse interface includes a menu bar with 'Test', 'Edit', 'Add', 'Tools', and 'Execution Log'. The test results are displayed in a table with columns for test ID, input parameters, and the result. Below the table is a detailed view of the 'tests_fitnesse_fixture.Test_addBook' table.

Test ID	Input Parameters	Result	Expected Output	Actual Output
2	1,Author3,Title3,ISBN3,Publisher3	true	Title: Title3 Author: Author3 ISBN: ISBN3 Publisher: Publisher3	Title: Title3 Author: Author3 ISBN: ISBN3 Publisher: Publisher3
3	3,Author1,Title1,ISBN1,Publisher1,Actor1	true	Title: Title1 Author: Author1 ISBN: ISBN1 Publisher: Publisher1 Actor: Actor1	Title: Title1 Author: Author1 ISBN: ISBN1 Publisher: Publisher1 Actor: Actor1
3	3,Author1,Title1,ISBN1,Publisher1,Actor1	false	Title: Title1 Author: Author1 ISBN: ISBN1 Publisher: Publisher1 Actor: Actor1	null
4	3,Author2,Title2,ISBN2,Publisher2,Actor2	true	Title: Title2 Author: Author2 ISBN: ISBN2 Publisher: Publisher2 Actor: Actor2	Title: Title2 Author: Author2 ISBN: ISBN2 Publisher: Publisher2 Actor: Actor2
5	3,Author4,Title4,ISBN4,Publisher4,Actor4	true	Title: Title4 Author: Author4 ISBN: ISBN4 Publisher: Publisher4 Actor: Actor4	Title: Title4 Author: Author4 ISBN: ISBN4 Publisher: Publisher4 Actor: Actor4
5	4,Author4,Title4,ISBN4,Publisher4,Actor4	false	Title: Title4 Author: Author4 ISBN: ISBN4 Publisher: Publisher4 Actor: Actor4	null

datatitles	databooks	number	index	addBook?	result
0,ISBN1	0,1	1	0	true	true
0,ISBN1	0,1	0	0	false	false
0,ISBN1	0,2	2	0	true	true
0,ISBN5	0,2	0	0	false	false
2,ISBN4,Actor4	0,1	1	1	true	true
2,ISBN4,Actor4	0,1	0	1	false	false
2,ISBN4,Actor4	0,2	2	1	true	true
2,ISBN2,Actor4	0,2	0	1	false	false
4,ISBN4,Actor4	0,1	0	1	false	false

Front Page | User Guide | root (for global ipath's, etc.) | Press '?' for keyboard shortcuts | Plugins | Contact | (edit)

3.13. Wykonanie testu akceptacyjnego dodawania rezerwacji przez testowaną aplikację – dalej podano definicję klasy **Test_addReservation** zawierającej kod do testowania funkcji **addReservation** klasy **Facade**. Kolorem czerwonym zaznaczono nazwy metod i atrybutów, zastosowanych przy budowie tablicy decyzyjnej testu na stronie **addReservation**.

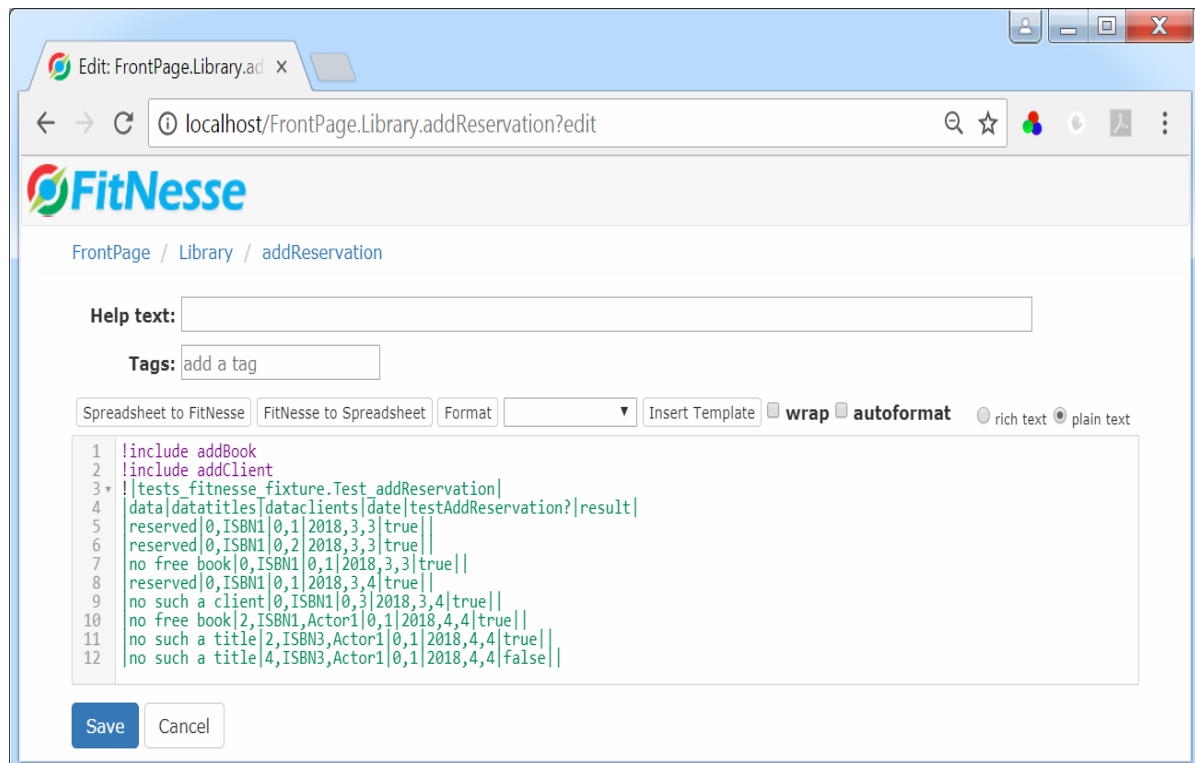
```
package tests_fitnessse_fixture;

import fit.ColumnFixture;
import java.time.LocalDate;
import java.util.IllegalFormatCodePointException;

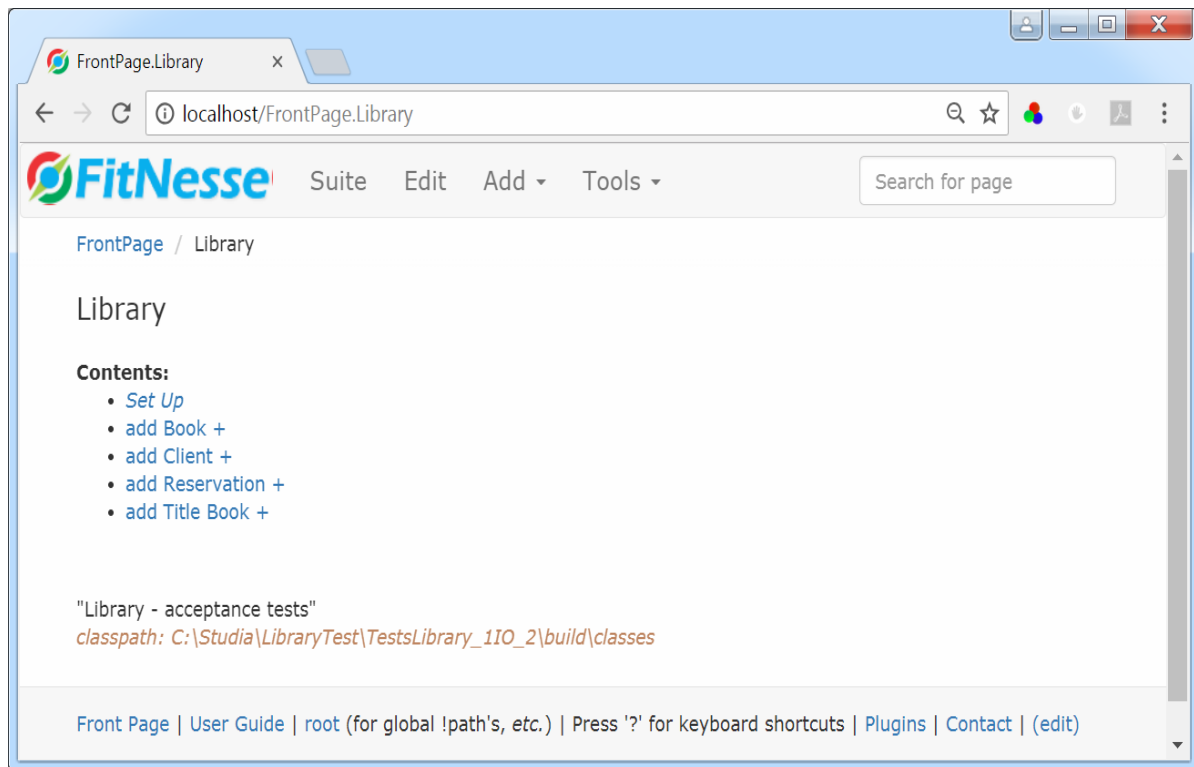
public class Test_addReservation extends ColumnFixture {
    String datatitles[], dataclients[], result, data;
    int date[];

    public boolean testAddReservation() {
        try {
            result=SetUp.facade.addReservation(datatitles, dataclients, LocalDate.of(date[0],date[1],date[2]));
        } catch (IllegalFormatCodePointException e) {
            result="no such a title";
            return false;
        }
        return data.equals(result); }
}
```

Definicja strony **addReservation** do testowania dodawania nowej rezerwacji (rysunek poniżej) opiera się na czynnościach takich samych, jak przy tworzeniu testu dodawania klienta w p.3.10. W treści strony należy uruchomić dwa poprzednie testy dotyczące dodawania książki i klienta, aby na podstawie wprowadzonych danych podczas testowania wprowadzania klientów i książek wykonać testy akceptacyjne procesu dodawania rezerwacji książek. Dokonano tego dodając znaczniki **!include** i podając nazwy stron testowych: **addBook** oraz **addClient**. Dane wzorcowe testu pobrano z klasy **Data**.



Poniżej podano widok strony głównej **Library** po dodaniu nowej strony testowej **addReservation**.



Poniżej pokazano widok strony testowej **addReservation** po uruchomieniu ze strony **Library**.

The screenshot shows a web browser window with the address bar displaying `localhost/ FrontPage.Library.addReservation`. The page content is organized into several sections, each containing a table of test results:

- addTitleBook:** A table with columns: number, datatitles, addTitleBook?, data, result. It contains 10 rows of test data.
- addBook:** A table with columns: datatitles, databooks, number, Index, addBook?, result. It contains 10 rows of test data.
- addClient:** A table with columns: number, dataclients, addClient?, data, result. It contains 3 rows of test data.
- addReservation:** A table with columns: data, datatitles, dataclients, date, testAddReservation?, result. It contains 8 rows of test data.

At the bottom of the browser window, there is a footer with the text: `Front Page | User Guide | root (for global paths, etc.) | Press '!' for keyboard shortcuts | Plugins | Contact | (edit)`

Wykonanie testu dodawania nowej rezerwacji po uruchomieniu testu za pomocą pozycji **Test z Menu Bar** strony **addReservation** pokazano na rysunku poniżej.

Test Results: FrontPage.Li x

localhost/FrontPage.Library.addReservation?test

FitNesse Test Edit Add - Tools - Execution Log

FrontPage / Library / addReservation

Test Pages: 1 right, 0 wrong, 0 ignored, 0 exceptions Assertions: 29 right, 0 wrong, 0 ignored, 0 exceptions (1,030 seconds)

Test System: fit:fit.FitServer

Included page: FrontPage.Library.Setup (edit)

Included page: addBook (edit)

Included page: addTitleBook (edit)

number	dataTitle	addTitleBook?	data	result
0	1,Auflor1,Ti0c1,ISBN1,Publisher1	true	Title: Ti0c1 Author: Auflor1 ISBN: ISBN1 Publisher: Publisher1	Title: Ti0c1 Author: Auflor1 ISBN: ISBN1 Publisher: Publisher1
0	1,Auflor1,Ti0c1,ISBN1,Publisher1	false	Title: Ti0c1 Author: Auflor1 ISBN: ISBN1 Publisher: Publisher1	null
1	1,Auflor2,Ti0c2,ISBN2,Publisher2	true	Title: Ti0c2 Author: Auflor2 ISBN: ISBN2 Publisher: Publisher2	Title: Ti0c2 Author: Auflor2 ISBN: ISBN2 Publisher: Publisher2
2	1,Auflor3,Ti0c3,ISBN3,Publisher3	true	Title: Ti0c3 Author: Auflor3 ISBN: ISBN3 Publisher: Publisher3	Title: Ti0c3 Author: Auflor3 ISBN: ISBN3 Publisher: Publisher3
3	3,Auflor1,Ti0c1,ISBN1,Publisher1,Astor1	true	Title: Ti0c1 Author: Auflor1 ISBN: ISBN1 Publisher: Publisher1 Actor: Astor1	Title: Ti0c1 Author: Auflor1 ISBN: ISBN1 Publisher: Publisher1 Actor: Astor1
3	3,Auflor1,Ti0c1,ISBN1,Publisher1,Astor1	false	Title: Ti0c1 Author: Auflor1 ISBN: ISBN1 Publisher: Publisher1 Actor: Astor1	null
4	3,Auflor2,Ti0c2,ISBN2,Publisher2,Astor2	true	Title: Ti0c2 Author: Auflor2 ISBN: ISBN2 Publisher: Publisher2 Actor: Astor2	Title: Ti0c2 Author: Auflor2 ISBN: ISBN2 Publisher: Publisher2 Actor: Astor2
5	3,Auflor4,Ti04,ISBN4,Publisher4,Astor4	true	Title: Ti04 Author: Auflor4 ISBN: ISBN4 Publisher: Publisher4 Actor: Astor4	Title: Ti04 Author: Auflor4 ISBN: ISBN4 Publisher: Publisher4 Actor: Astor4
5	4,Auflor4,Ti04,ISBN4,Publisher4,Astor4	false	Title: Ti04 Author: Auflor4 ISBN: ISBN4 Publisher: Publisher4 Actor: Astor4	null

dataTitle	dataBook	number	index	addBook?	result
0,ISBN1	0,1	1	0	true	true
0,ISBN1	0,1	0	0	false	false
0,ISBN1	0,2	2	0	true	true
0,ISBN3	0,2	0	0	false	false
2,ISBN4,Astor4	0,1	1	1	true	true
2,ISBN4,Astor4	0,1	0	1	false	false
2,ISBN4,Astor4	0,2	2	1	true	true
2,ISBN2,Astor4	0,2	0	1	false	false
4,ISBN4,Astor4	0,1	0	1	false	false

Included page: addClient (edit)

number	dataClient	addClient?	data	result
0	1,Client1,1	true	Client{ numberCard=1, name=Client1 reservations=[]}	Client{ numberCard=1, name=Client1 reservations=[]}
1	1,Client2,2	true	Client{ numberCard=2, name=Client2 reservations=[]}	Client{ numberCard=2, name=Client2 reservations=[]}
1	1,Client2,1	false	Client{ numberCard=2, name=Client2 reservations=[]}	null

data	dataTitle	dataClient	date	testAddReservation?	result
reserved	0,ISBN1	0,1	2018.3.3	true	reserved
reserved	0,ISBN1	0,2	2018.3.3	true	reserved
no free book	0,ISBN1	0,1	2018.3.3	true	no free book
reserved	0,ISBN1	0,1	2018.3.4	true	reserved
no such a client	0,ISBN1	0,3	2018.3.4	true	no such a client
no free book	2,ISBN1,Astor1	0,1	2018.4.4	true	no free book
no such a title	2,ISBN2,Astor1	0,1	2018.4.4	true	no such a title
no such a title	4,ISBN2,Astor1	0,1	2018.4.4	false	no such a title

Front Page | User Guide | root (for global ipath's, etc.) | Press F7 for keyboard shortcuts | Plugins | Contact | (edit)