Enterprise JavaBeans (EJB)

Celem tego zestawu ćwiczeń jest zapoznanie z sesyjnymi komponentami Enterprise JavaBeans. Zilustrowane będą różnice między komponentami stanowymi i bezstanowymi. Pokazane będzie tworzenie klientów aplikacyjnych i webowych.

Ćwiczenia zostały przygotowane dla środowiska NetBeans 7.0 i zintegrowanego z NetBeans serwera GlassFish.

Ćwiczenie 1

W tym ćwiczeniu zostanie utworzony projekt w środowisku NetBeans, składający się z trzech modułów.

1. Uruchom środowisko NetBeans.

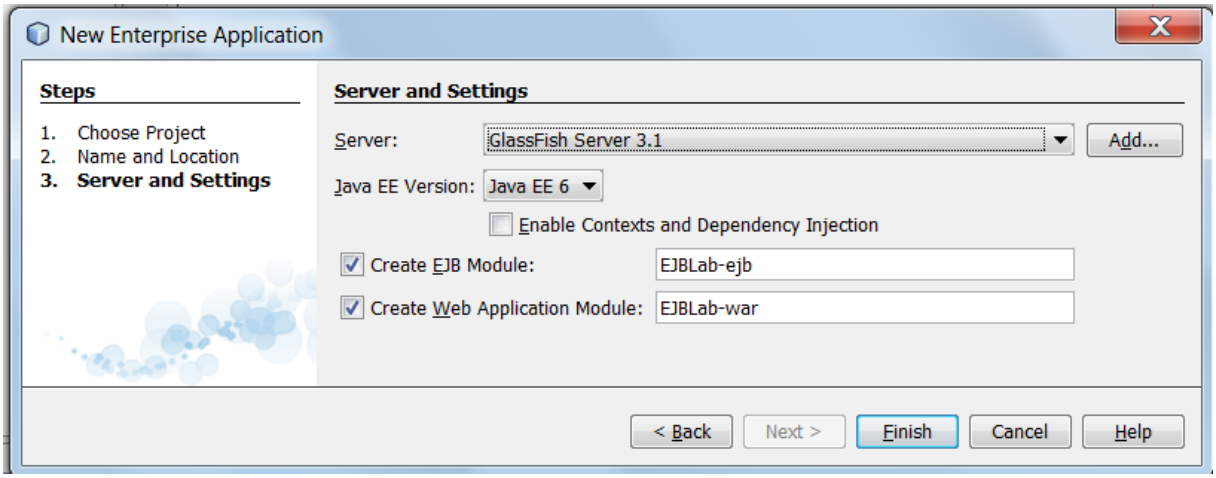
2. Utwórz nowy projekt typu Enterprise Application:

a) Wybierz z menu opcję File->New Project.

b) W pierwszym kroku kreatora jako typ projektu wybierz Enterprise Application z kategorii Java EE.

c) W drugim kroku kreatora jako nazwę projektu podaj „EJBLab”, a pozostałe ustawienia w tym oknie pozostaw domyślne.

d) W ostatnim kroku kreatora wybierz najnowszą dostępną wersję serwera GlassFish (GlassFish Server 3.1 w NetBeans 7.0.1) i najnowszą dostępną wersję Java EE (Java EE 6 w NetBeans 7.0.1). Upewnij się że zaznaczone jest tworzenie wszystkich dwóch typów modułów: EJB i webowego. Pozostaw zaproponowane przez kreator nazwy modułów.



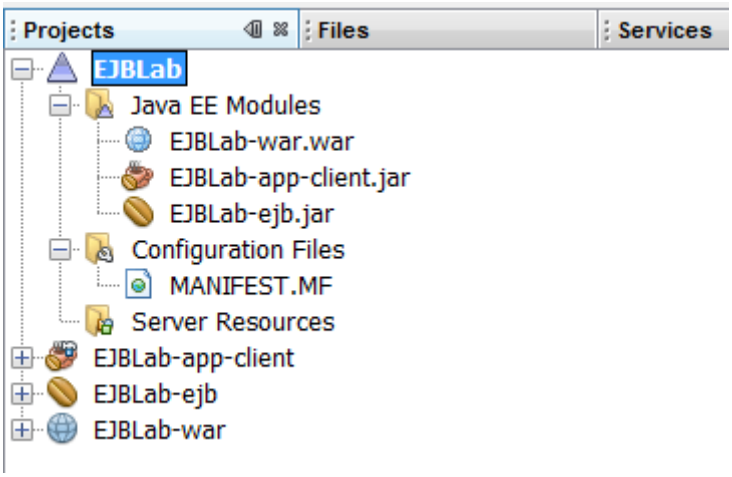
e) Ponownie wybierz z menu opcję File->New Project.

f) W pierwszym kroku kreatora jako typ projektu wybierz Enterprise Application Client z kategorii Java EE.

g) W drugim kroku kreatora jako nazwę projektu podaj „EJBLab-app-client”, a pozostałe ustawienia w tym oknie pozostaw domyślne.

h) W ostatnim kroku kreatora z listy dostępnej dla opcji Add to Enterprise Application wybierz naszą aplikację „EJBLab”. Serwer i wersję Java EE wybierz takie jak wcześniej dla głównego projektu. Pozostałe opcje pozostaw domyślne.

i) Po zakończeniu działania kreatora obejrzyj organizację aplikacji w oknie Projects. Zwróć uwagę, że w środowisku NetBeans moduły składowe aplikacji Enterprise Application stanowią odrębne projekty, a główny projekt łączy je w całość (inne środowiska IDE mogą stosować inną organizację kodu).



Uwaga: Wcześniejsze wersje środowiska NetBeans (6.x) umożliwiały tworzenie klienta aplikacyjnego jako modułu aplikacji Java EE w ramach wywołania kreatora dla głównej aplikacji, analogicznie jak klienta webowego.

Ćwiczenie 2

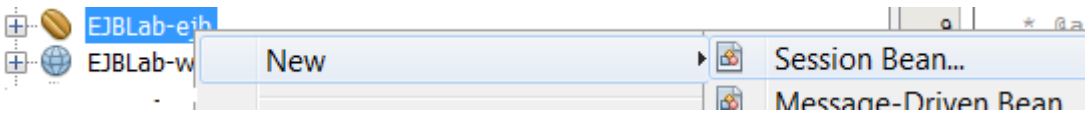
W tym ćwiczeniu utworzone zostaną utworzone 2 sesyjne komponenty EJB: jeden bezstanowy, a drugi stanowy. Komponenty będą implementowały funkcję licznika – każda instancja będzie zliczała liczbę wywołań metody usługowej.

1. Utwórz nowy projekt typu Java Class Library z kategorii Java. Nazwij go „EJBLab-remote-client-lib”.

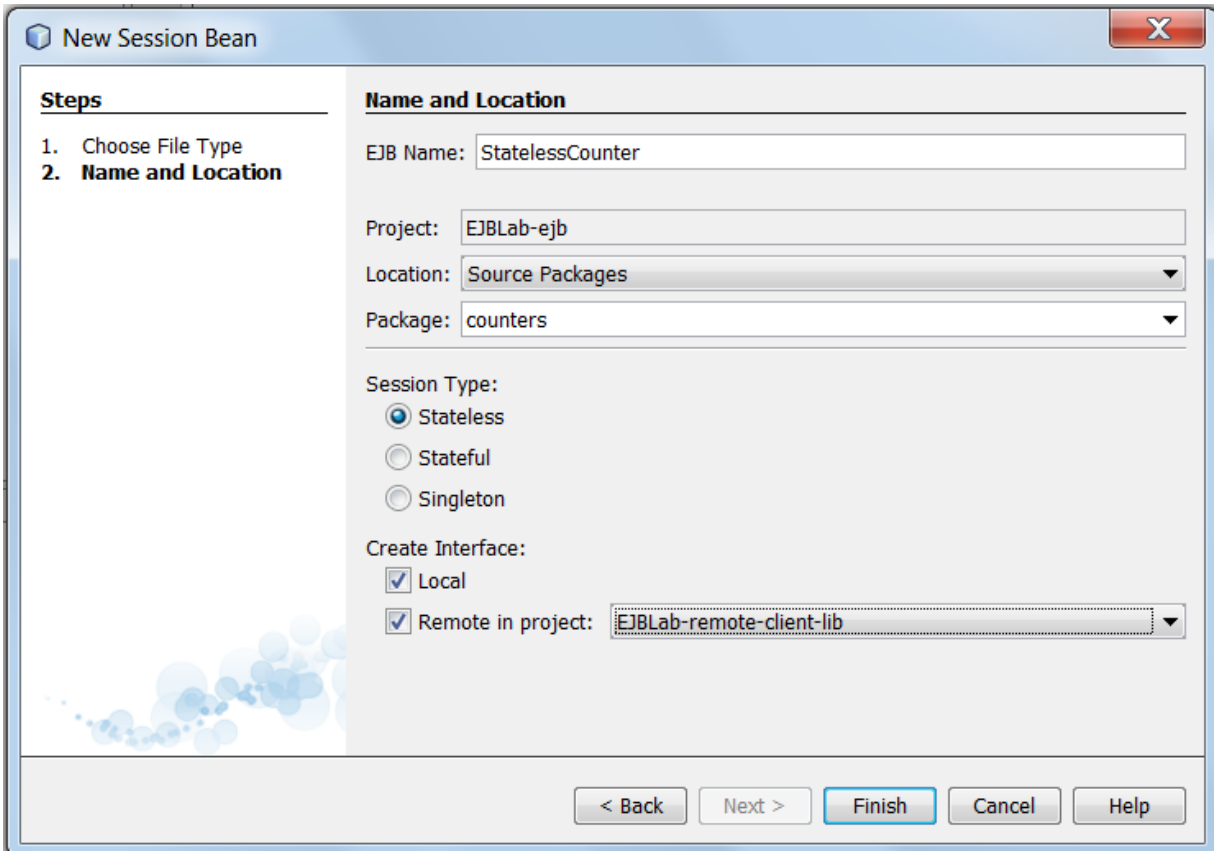
Uwaga: Projekt ten będzie zawierał interfejs zdalny komponentu EJB, który zostanie utworzony w kolejnym punkcie ćwiczenia. NetBeans 7 promuje praktykę umieszczania interfejsów zdalnym w odrębnym projekcie, co później ułatwi korzystanie z nich w tworzonych aplikacjach zdalnych klientów.

2. W projekcie modułu EJB utwórz bezstanowy sesyjny EJB:

a) W oknie Projects wybierz z menu kontekstowego dla węzła projektu modułu EJB opcję New->Session Bean.



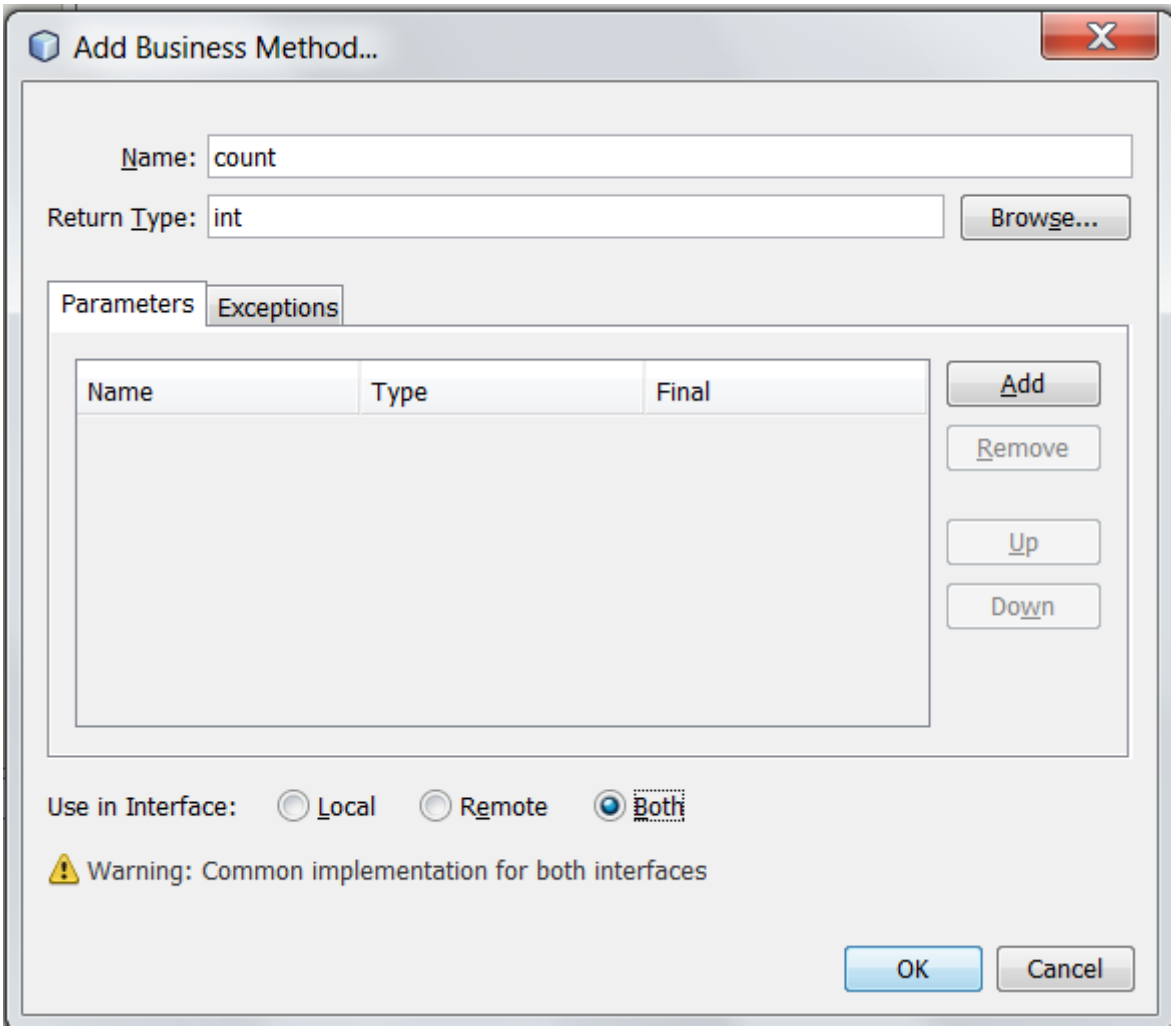
b) Jako nazwę komponentu podaj „StatelessCounter”, a jako nazwę pakietu „counters”. Jako typ komponentu wskaż: bezstanowy. Zaznacz opcję tworzenia zarówno interfejsu zdalnego jak i lokalnego. Wskaż, że interfejs zdalny ma być umieszczony w utworzonym w poprzednim punkcie ćwiczenia projekcie biblioteki.



c) obejrzyj kod wygenerowanych przez kreator 3 plików źródłowych (2 interfejsy i klasa komponentu). Zwróć uwagę na adnotacje związane z technologią EJB.

3. Dodaj do utworzonego komponentu EJB prywatne pole counter typu int inicjowane na 0.

4. Dodaj do utworzonego komponentu EJB publiczną bezargumentową metodę count() zwracającą wartość int. Użyj sposobu zaproponowanego w komentarzu umieszczonym przez kreator w klasie komponentu (Insert Code -> Add Business Method). Spraw aby metoda była dostępna poprzez interfejs zdalny i lokalny.

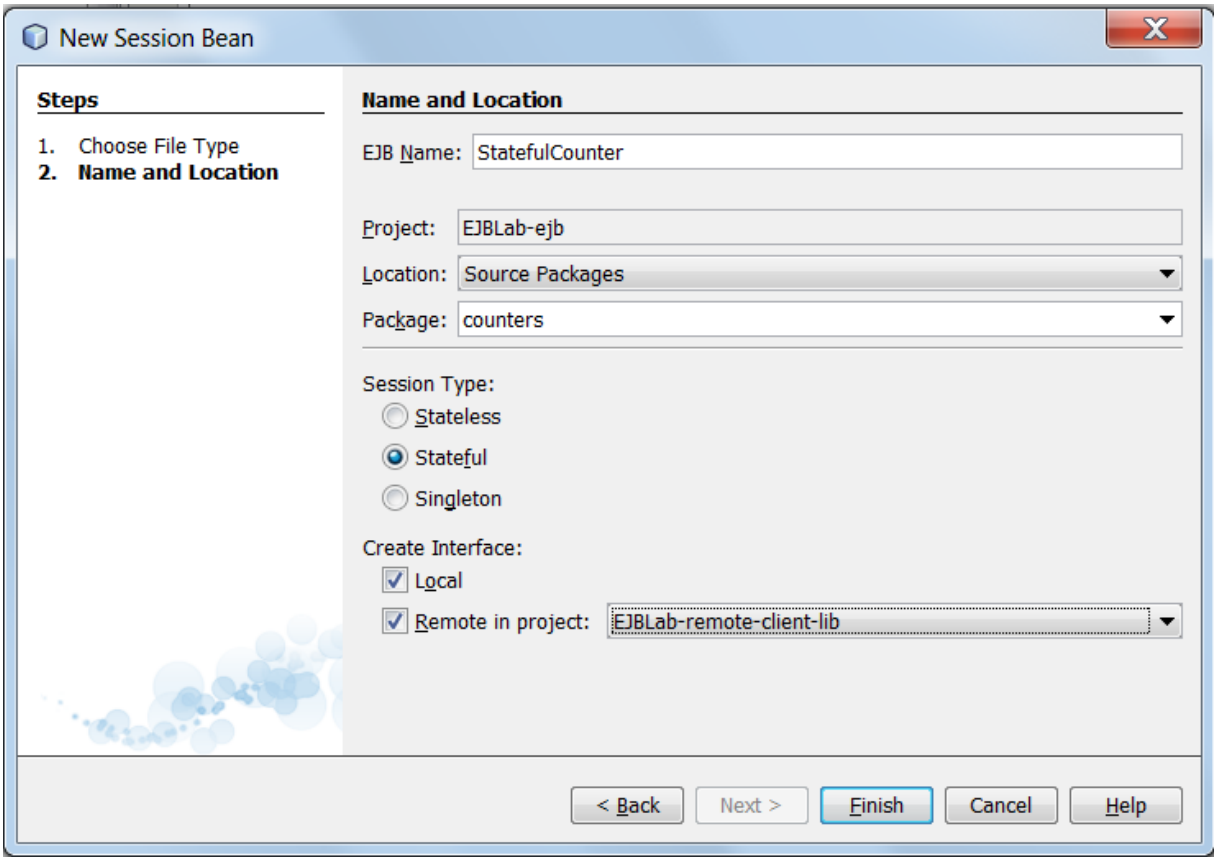


Uwaga: Kreator ostrzega, że udostępnienie tej samej metody przez interfejs zdalny i lokalny jest niezalecane i potencjalnie może być źródłem problemów np. ze względu na różnice w sposobie przekazywania parametrów. W naszej prostej aplikacji możemy zignorować to ostrzeżenie.

5. Zmień kod metody wygenerowany przez kreator, tak aby metoda zwiększała licznik o 1 i po zwiększeniu zwracała wartość licznika jako wynik.

6. Obejrzyj kod interfejsu lokalnego i zdalnego i upewnij się, że oba zawierają metodę count().

7. Utwórz w tym samym projekcie (module) stanowy sesyjny komponent EJB. Nazwij go „StatefulCounter”, a jako nazwę pakietu podaj „counters”. Zaznacz opcję tworzenia zarówno interfejsu zdalnego jak i lokalnego, umieszczając zdalny interfejs w projekcie biblioteki.



8. Powtórz kroki 3-6 dla komponentu stanowego, tak aby podobnie jak bezstanowy pełnił funkcję licznika wywołań.

Ćwiczenie 3

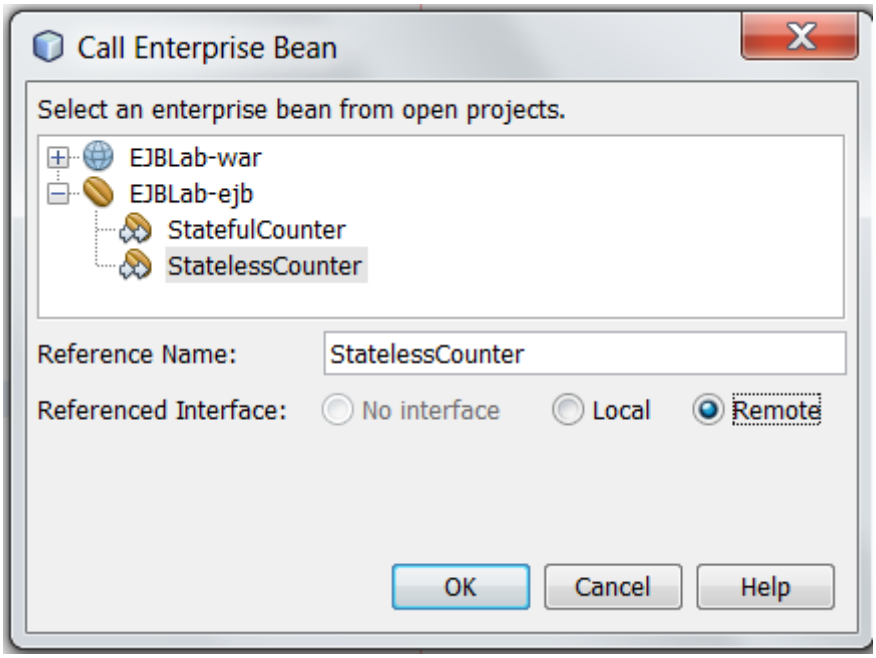
W tym ćwiczeniu utworzone zostaną utworzeni aplikacyjni klienci komunikujący się z sesyjnymi komponentami przez interfejs zdalny. W oparciu o tych klientów przeprowadzony zostanie eksperyment ilustrujący różnicę między stanowymi i bezstanowymi sesyjnymi EJB.

1. Przejdź do edycji głównej klasy w projekcie modułu klienta aplikacyjnego.

2. Dodaj w klasie głównej klienta aplikacyjnego odwołanie do komponentu EJB:

a) Wybierz z menu kontekstowego opcję Insert Code -> Call Enterprise Bean.

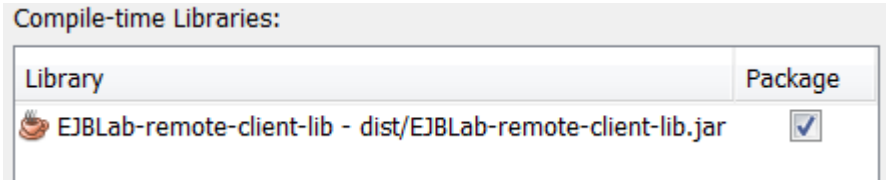
b) Z wyświetlonej listy dostępnych komponentów wybierz komponent bezstanowy. Pozostaw bez zmian zaproponowaną nazwę referencji. Zaznacz, że komunikacja będzie realizowana przez interfejs zdalny.



UWAGA: Jeśli po rozwinięciu gałęzi projektu EJB nie pojawią się do wyboru dwa utworzone wcześniej sesyjne komponenty EJB, zapisz zmiany i zrestartuj środowisko NetBeans.

c) Zwróć uwagę na dodane przez kreator wstrzyknięcie zależności dla komponentu EJB do statycznej zmiennej w klasie.

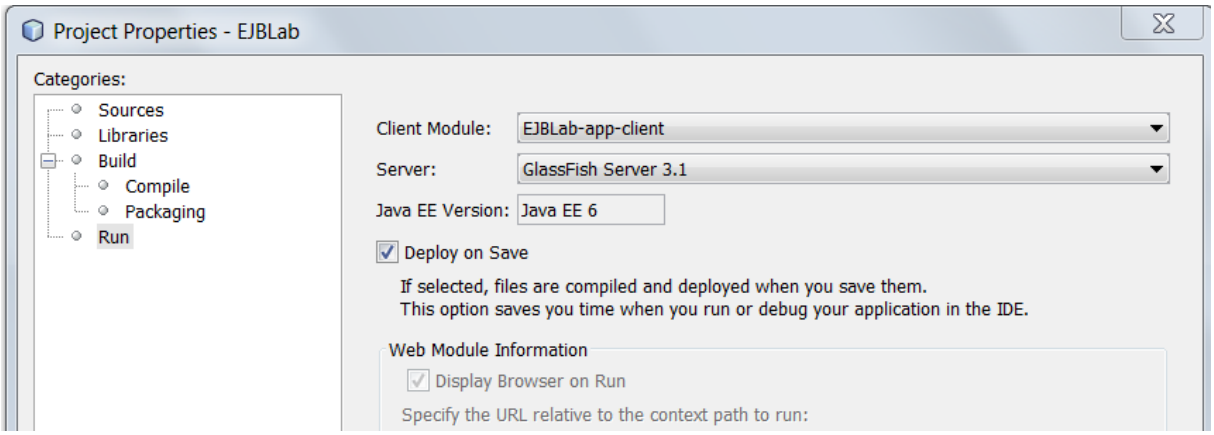
3. Podejrzyj właściwości projektu klienta aplikacyjnego. Odszukaj dodane przez wywołany przed chwilą kreator wywołania komponentu EJB ustawienie dzięki któremu projekt klienta aplikacyjnego widzi interfejsy zdalne komponentów EJB.



4. Umieść w metodzie main() poniższy kod, wywołujący metodę usługową komponentu w jednosekundowych odstępach czasowych:

```
System.out.println( statelessCounter.count(  ) );
try {Thread.sleep(1000);} catch (InterruptedException e) {};
System.out.println( statelessCounter.count(  ) );
try {Thread.sleep(1000);} catch (InterruptedException e) {};
System.out.println( statelessCounter.count(  ) );
try {Thread.sleep(1000);} catch (InterruptedException e) {};
System.out.println( statelessCounter.count(  ) );
try {Thread.sleep(1000);} catch (InterruptedException e) {};
System.out.println( statelessCounter.count(  ) );
```

5. Upewnij się, że moduł klienta aplikacyjnego jest wybrany jako moduł klienta naszej aplikacji. W tym celu przejdź do właściwości głównego projektu (EJBLab) i podejrzyj zakładkę Run. Jeśli jako Client Module nie jest wybrany EJBLab-app-client, to go wybierz z listy.



6. Uruchom główny projekt aplikacji. Zaobserwuj wynik działania na konsoli.

7. Uruchom współbieżnie aplikację 2 razy poprzez wywołanie opcji Run w możliwie krótkim odstępie czasu. Zinterpretuj zaobserwowane wyświetlone wartości licznika.

8. Zmodyfikuj (ręcznie) kod klasy głównej klienta aplikacyjnego, tak aby tym razem korzystała ze stanowego sesyjnego EJB.

9. Uruchom główny projekt aplikacji. Zaobserwuj wynik działania na konsoli.

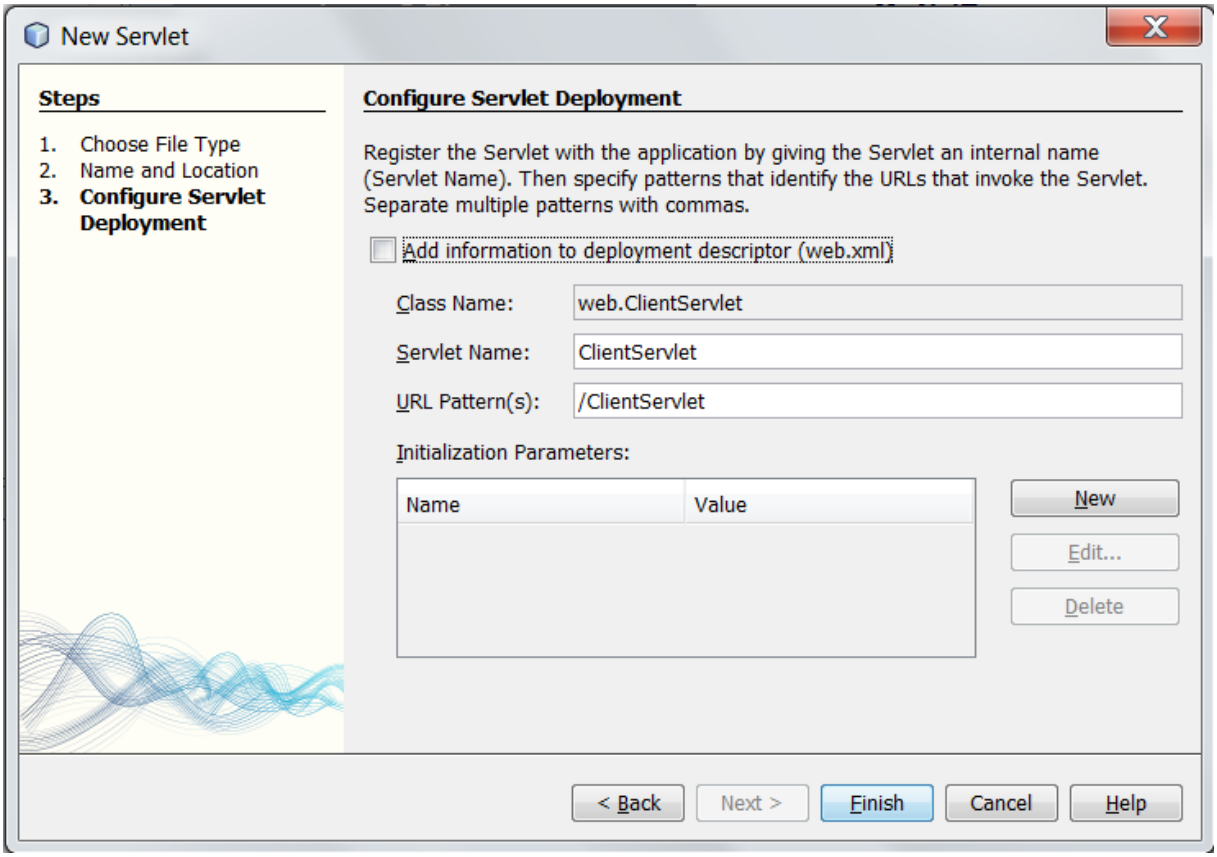
10. Uruchom współbieżnie aplikację 2 razy poprzez wywołanie opcji Run w możliwie krótkim odstępie czasu. Zinterpretuj zaobserwowane wyświetlone wartości licznika.

Ćwiczenie 4

W tym ćwiczeniu zostanie utworzony serwlet wywołujący metody sesyjnych komponentów EJB poprzez interfejs lokalny.

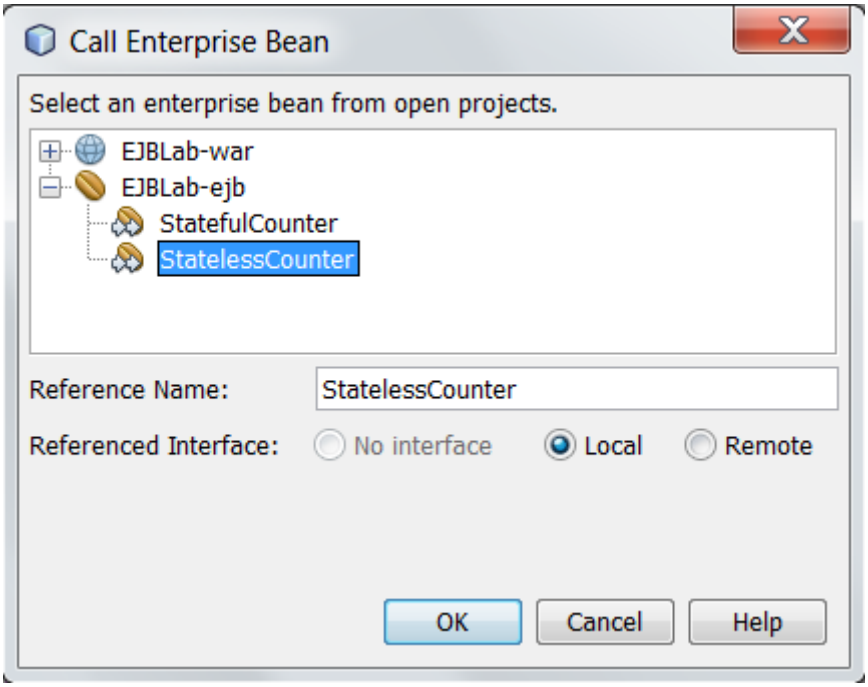
1. Zaznacz moduł klienta webowego jako moduł klienta naszej aplikacji. W tym celu przejdź do właściwości głównego projektu (EJBLab) i podejrzyj zakładkę Run. Jeśli jako Client Module nie jest wybrany EJBLab-war, to go wybierz z listy.

2. Utwórz w projekcie webowym (EJBLab-war) nowy serwlet. Jako jego nazwę podaj „ClientServlet” a jako nazwę pakietu „web”. Pozostałe ustawienia serwletu pozostaw domyślne (w szczególności nie zaznaczaj opcji dodawania informacji do pliku web.xml).



Obejrzyj kod klasy wygenerowanego serwletu zwracając uwagę na adnotację specyfikującą ścieżkę URL, która będzie wywoływać serwlet.

3. Dodaj w kodzie serwletu (na poziomie klasy) referencje do obu utworzonych wcześniej komponentów EJB. Wykorzystaj do tego opcję Insert Code->Call Enterprise Bean. Zaakceptuj proponowane nazwy referencji. Dla obu komponentów zwróć uwagę, aby odwołanie było poprzez interfejs lokalny.



Obejrzyj sposób w jaki serwletowi udostępnione zostały referencje do stanowego i bezstanowego komponentu EJB. Komponent stanowy powinien być jawnie wyszukiwany przez interfejs JNDI, gdyż wstrzykiwanie zależności w serwlecie dostępne jest tylko na poziomie klasy serwletu, a nie wewnątrz metod – co okaże się konieczne w dalszej części ćwiczenia. Zwróć uwagę na dodaną przez kreator metodę pomocniczą realizującą lookup JNDI. Przeanalizuj nazwę globalną przez którą komponent jest identyfikowany w JNDI.

4. Rozwiń kod wygenerowanego serwletu. Przeanalizuj zależności między metodami doGet(), doPost() i processRequest().

5. Zmodyfikuj kod metody processRequest() tak aby generowała dokument HTML prezentujący stan obu liczników reprezentowanych przez komponenty EJB.

6. Uruchom aplikację. W dwóch różnych przeglądarkach wprowadź adres URL wywołujący serwlet (http://localhost:8080/EJBLab-war/ClientServlet). Odśwież stronę kilka razy w każdej z przeglądarek. Spróbuj wyjaśnić zachowanie stanowego licznika.

7. W przypadku odwołania z serwletu do stanowego sesyjnego EJB należy zapewnić aby każda sesja miała swoją własną referencję do EJB. Zastąpienie wstrzykiwania zależności przez jawne wyszukanie komponentu przez JNDI nie rozwiązuje problemu jeśli wyszukana referencja będzie przypisana do zmiennej instancji serwletu (współdzielonej przez wszystkie wątki serwletu, czyli wszystkie żądania).

a) Przenieś deklarację referencji do stanowego sesyjnego EJB z poziomu serwletu do metody processRequest() (wraz z wywołaniem metody realizującej wyszukanie komponentu przez JNDI). Zapisz zmiany, uruchom i przetestuj aplikację. Jak możesz wyjaśnij jej aktualne działanie?

b) zmodyfikuj kod metody processRequest() serwletu tak aby referencja do komponentu stanowego była przechowywana w sesji HTTP (obiekt HttpSession) na czas kolejnych wywołań. (Wyszukanie w JNDI powinno mieć miejsce tylko wtedy gdy nie ma w zasięgu sesji wcześniej wyszukanej referencji.)

c) Uruchom aplikację. W dwóch różnych przeglądarkach wprowadź adres URL wywołujący serwlet (http://localhost:8080/EJBLab-war/ClientServlet). Odśwież stronę kilka razy w każdej z przeglądarek.

Ćwiczenie 5

W tym ćwiczeniu zostanie utworzona strona JSF wywołująca metodę stanowego sesyjnego komponentu EJB poprzez interfejs lokalny.

1. Dodaj w module webowym nową stronę JSF opartą o Facelets. Nazwij ją „index”.

2. Utwórz w module webowym nowy komponent zarządzany. Dodaj w klasie komponentu zarządzanego metodę getCurrentNumber() zwracającą wartość licznika pobraną ze stanowego sesyjnego EJB.

3. Na stronie JSF umieść komponent do wyświetlania stanu licznika udostępnianego przez komponent zarządzany.

4. Uruchom aplikację. Przetestuj działanie strony otwierając i odświeżając ją współbieżnie w dwóch różnych przeglądarkach. Powtórz eksperyment ustawiając komponentowi zarządzanemu zasięg żądania, sesji i aplikacji. Który zasięg jest odpowiedni do pośredniczenia w dostępie do stanowego sesyjnego EJB?