**Лекция 14**

**Второй пример**

Одна из основных идей, лежащих в основе технологии Web -серви-сов, состоит в независимости разрабатываемых на ее основе приложений от используемой платформы. Поэтому, в качестве дополнения к предыдущему примеру, мы реализуем второго клиента для того же самого Web -сервиса, но на основе совершенно другой технологии. В качестве клиента мы будем использовать стандартное Web -приложение, состоящее из набора HTML -страниц с небольшими вкраплениями jsp -кода. Допустим, для автоматизируемой нами сети ресторанов нужно создать Web -сайт, на котором клиенты могли бы просмотреть текущий баланс своих карт, а сотрудники могли бы выполнить некоторые действия с системой - например, создать новую карту. Соответствующий код, реализующий такую функциональность, представлен ниже.

Поскольку мы не ставим своей целью рассмотреть в подробностях процесс Web -программирования, остановимся только на основных деталях. По сути, наше приложение состоит из нескольких jsp -страниц. Поскольку мы разделили страницы, содержащие операции запроса баланса и создания новой карты, их получилось четыре. Для того чтобы jsp -код смог работать с Web -сервисом, необходимо, как и в случае с предыдущим клиентом, сгенерировать вспомогательные классы с помощью утилиты wsimport и разместить их в соответствующей папке ( classes ) Web -приложения.

Итак, реализуем две jsp -страницы, которые содержат формы для ввода параметров: одна для метода просмотра баланса (вводится один параметр - номер просматриваемой карты), вторая для операции заведения новой карты (вводится два параметра - номер карты и персона). Еще две jsp -страницы загружаются первыми при нажатии соответствующих кнопок в форме и содержат код, вызывающий методы Web -сервиса.

**Страница ShowBalance.jsp**

Содержит форму параметров для вызова метода getCard Web -сервиса ([пример 12.5](https://www.intuit.ru/studies/courses/633/489/lecture/11090?page=3#example.12.5)). Вызывает showbalanceresponse.jsp.

<html>

<head><title>BillingService</title></head>

<h2>Enter card number</h2>

<form method="get">

<input type="text" name="cardnumber" size="25">

<p></p>

<input type="submit" value="Submit">

<input type="reset" value="Reset">

</form>

<%

String cardnumber = request.getParameter("cardnumber");

if ( cardnumber != null && cardnumber.length() > 0 ) {

%>

<%@include file="showbalanceresponse.jsp" %>

<%

}

%>

</body>

</html>

Листинг 12.5. Страница ShowBalance.jsp

**Страница showbalanceresponse.jsp**

Содержит код для обращения к Web -сервису ([пример 12.5](https://www.intuit.ru/studies/courses/633/489/lecture/11090?page=3#example.12.5)), для чего подключает классы **com.asw.ws.ex1.endpoint.BillingService, com.asw.ws.ex1. endpoint.Billing, com.asw.ws.ex1.endpoint.Card**,которые были автоматически сгенерированы утилитой wsimport на основании анализа WSDL файла Web -сервиса. Эти классы должны располагаться в папке classes Web -приложения. Вызывает метод Web -сервиса getCard, передавая ему в качестве аргумента значение, введенное в форме. Поля возвращенного объекта выводятся на странице.

<%@ page import="com.asw.ws.ex1.endpoint.BillingService,

com.asw.ws.ex1.endpoint.Billing, com.asw.ws.ex1.endpoint.Card" %>

<%

Card resp = null;

try {

Billing billing = new BillingService().getBillingPort();

resp = billing.getCard(request.getParameter("cardnumber"));

} catch (Exception ex) { resp = new Card();

}

%>

<h2>

<font color="black">

<%=resp.getPerson()+"\t"+resp.getBalance()%>

</font></h2>

Листинг 12.6. Страница showbalanceresponse.jsp

**Страница AddCard.jsp**

Содержит форму параметров для вызова метода addNewCard Web -сервиса ([пример 12.7](https://www.intuit.ru/studies/courses/633/489/lecture/11090?page=3#example.12.7)) (в данном случае страница предназначена для создания одной карты за один раз, в качестве параметров вводятся номер карты и персона). Вызывает addcardresponse.jsp.

<html>

<head><title>BillingService</title></head>

<h2>Enter card number</h2>

<form method="get">

<input type="text" name="cardnumber"

size="25">

<input type="text" name="person"

size="25">

<p></p>

<input type="submit" value="Submit">

<input type="reset" value="Reset">

</form>

<%

String cardnumber = request.getParameter("cardnumber");

String person = request.getParameter("person");

if ( cardnumber != null && cardnumber.length() > 0 &&

person != null && person.length() > 0) {

%>

<%@include file="addcardresponse.jsp" %>

<%

}

%>

</body>

</html>

Листинг 12.7. Страница AddCard.jsp

**Страница addcardresponse.jsp**

Содержит код для обращения к Web -сервису ([пример 12.8](https://www.intuit.ru/studies/courses/633/489/lecture/11090?page=3#example.12.8)), для чего подключает классы **com.asw.ws.ex1.endpoint.BillingService, com.asw.ws.ex1. endpoint.Billing, com.asw.ws.ex1.endpoint.Card**,которые были автоматически сгенерированы утилитой wsimport на основании анализа WSDL файла Web -сервиса. Эти классы должны располагаться в папке **classes** Web -приложения. Вызывает метод Web -сервиса **addNewCard**,для чего создает объект класса **Card** и устанавливает его поля значениями, введенными в форме. Созданный объект добавляется в вектор, который передается в качестве аргумента методу **addNewCard**.

<%@ page import="com.asw.ws.ex1.endpoint.BillingService,

com.asw.ws.ex1.endpoint.Billing, com.asw.ws.ex1.endpoint.Card" %>

<%

Card card = new Card();

try {

Billing billing = new BillingService().getBillingPort();

String \_cardnumber = request.getParameter("cardnumber");

String \_person = request.getParameter("person");

card.setPerson(\_person);

card.setCardNumber(\_cardnumber);

java.util.Vector<Card> v = new java.util.Vector<Card>();

v.add(card);

billing.addNewCard(v);

} catch (Exception ex) {

}

%>

<h2> <font color="black">

<%=card.getPerson()+"\t"+card.getBalance()%>

</font>

</h2>

Листинг 12.8. Страница addcardresponse.jsp

**Структура каталога приложения**

Каталог Web -приложения будет иметь следующую структуру:

.\AddCard.jsp

.\addcardresponse.jsp

.\ShowBalance.jsp

.\showbalanceresponse.jsp

.\META-INF\MANIFEST.MF

.\WEB-INF\classes

.\WEB-INF\sun-web.xml

.\WEB-INF\web.xml

.\WEB-INF\classes\com\asw\ws\ex1\endpoint\AddMoney.class

.\WEB-INF\classes\com\asw\ws\ex1\endpoint\AddMoneyResponse.class

.\WEB-INF\classes\com\asw\ws\ex1\endpoint\AddNewCard.class

.\WEB-INF\classes\com\asw\ws\ex1\endpoint\AddNewCardResponse.class

.\WEB-INF\classes\com\asw\ws\ex1\endpoint\Billing.class

.\WEB-INF\classes\com\asw\ws\ex1\endpoint\BillingService.class

.\WEB-INF\classes\com\asw\ws\ex1\endpoint\Card.class

.\WEB-INF\classes\com\asw\ws\ex1\endpoint\CardOperation.class

.\WEB-INF\classes\com\asw\ws\ex1\endpoint\GetCard.class

.\WEB-INF\classes\com\asw\ws\ex1\endpoint\GetCardResponse.class

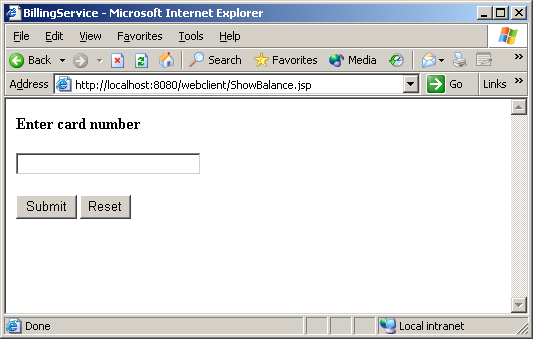
.\WEB-INF\classes\com\asw\ws\ex1\endpoint\ObjectFactory.class

.\WEB-INF\classes\com\asw\ws\ex1\endpoint\package-info.class

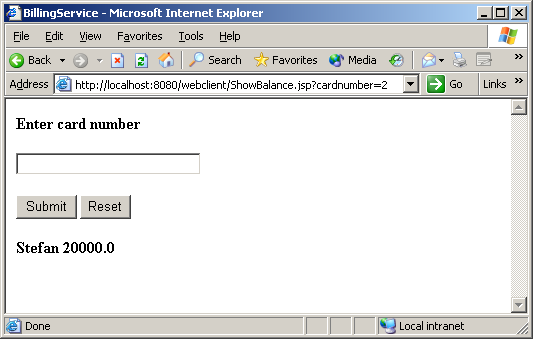
.\WEB-INF\classes\com\asw\ws\ex1\endpoint\ProcessOperation.class

.\WEB-INF\classes\com\asw\ws\ex1\endpoint\ProcessOperationResponse.class

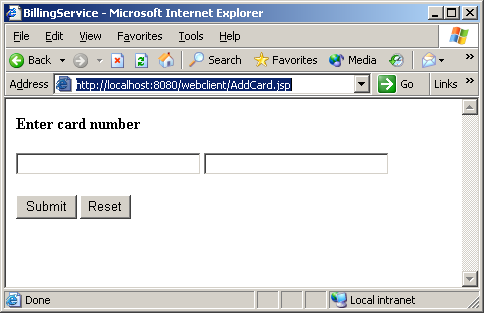
В корне приложения лежат разработанные нами jsp -страницы, в папке *WEB-INF* - файлы, содержащие необходимые для развертывания



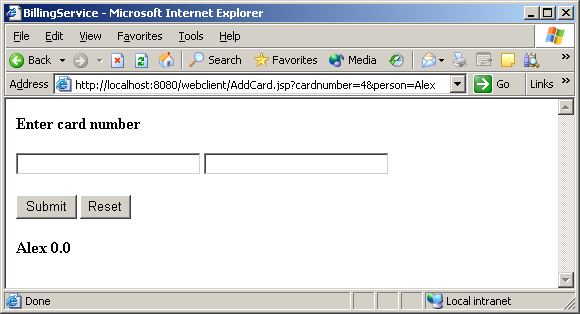
**Рис. 12.3.** Страница ShowBalance.jsp



**Рис. 12.4.** Результат запроса информации по карте



**Рис. 12.5.** Страница AddCard.jsp

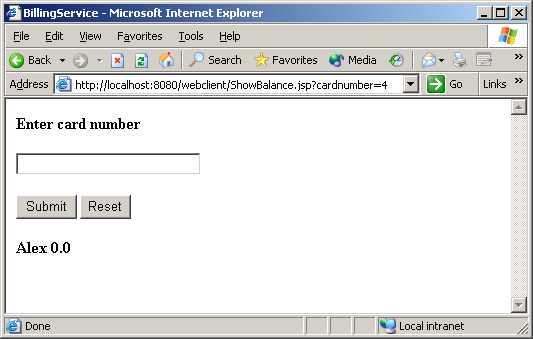


**Рис. 12.6.** Результат операции по вводу новой карты

на сервере и настройки нашего приложения параметры, а в папке *.\WEB-INF\ classes* - сгенерированные утилитой wsimport классы.

После развертывания приложения на сервере, обратившись по адресу <http://localhost:8080/webclient/ShowBalance.jsp> ( webclient - корневой раздел нашего приложения, определен в файле sun-web.xml ), мы увидим следующую страницу ([рис. 12.3](https://www.intuit.ru/studies/courses/633/489/lecture/11090?page=4#image.12.3)).

Введя номер карты и нажав кнопку *Submit*,мы получим информацию по карте с введенным номером ([рис 12.4](https://www.intuit.ru/studies/courses/633/489/lecture/11090?page=4#image.12.4)). Поскольку этот пример выполняется после того, как первый клиент выполнил 30000 операций над картами, значение баланса карт будет довольно велико.



**Рис. 12.7.** Результат запроса информации по вновь введенной карте

Для того чтобы создать новую карту, нужно загрузить другую страницу - <http://localhost:8080/webclient/AddCard.jsp> ([рис 12.5](https://www.intuit.ru/studies/courses/633/489/lecture/11090?page=4#image.12.5)).

Заполнив поля и нажав кнопку Submit, мы создадим новую карту в системе ([рис 12.6](https://www.intuit.ru/studies/courses/633/489/lecture/11090?page=4#image.12.6)), в чем можно убедиться, просмотрев информацию по ней с помощью первой страницы ([рис 12.7](https://www.intuit.ru/studies/courses/633/489/lecture/11090?page=4#image.12.7)).