

**Веб-служба, веб-сервис** (англ. **web service**) — идентифицируемая **веб**-адресом программная система со стандартизированными интерфейсами. **Веб**-службы могут взаимодействовать друг с другом и со сторонними приложениями посредством сообщений, основанных на определённых протоколах (SOAP, WSDL, XML-RPC и т. д.) и архитектурном стиле программирования REST.

- Веб-службы обеспечивают взаимодействие программных систем независимо от платформы. Например, Windows-C#-клиент может обмениваться данными с Java-сервером, работающим под [Linux](#). А возможно, что к веб-службе из Microsoft Azure, созданной на C#, обращается приложение на Android или iOS
- Веб-службы основаны на базе открытых стандартов и протоколов. Благодаря использованию [XML](#) достигается простота разработки и отладки веб-служб.

Примеры Вэб-служб:

Взаимодействие между авиакомпаниями и бюро путешествий. Первые предоставляют через веб-службы полезную информацию, которую вторые используют при поиске оптимальных предложений своим клиентам.

Веб-службы разворачиваются на [серверах приложений](#). Некоторые серверы приложений:

- [ColdFusion](#) от [Adobe](#)
- [DotGNU](#) от [GNU Project](#) (разработка остановлена)
- [GlassFish](#) — от компании [Oracle](#)
- [Google App Engine](#) — платформа для масштабируемых приложений, использующих инфраструктуру компании [Google](#)
- [IBM Lotus Notes](#) линейка ПО для организации совместной работы от [IBM](#)
- [JBoss](#) — компании [Red Hat](#)
- [Mono](#) — платформа разработки от [Xamarin](#) (ранее [Novell](#))
- [.NET Framework](#) серверы от [Microsoft](#)
- [Web Application Server](#) от [SAP](#) (является ключевой частью стека SAP NetWeaver)
- [WebLogic](#) от компании [Oracle](#) (продукт поглощённой [BEA Systems](#))
- [webMethods Integration Platform](#) от [Software AG](#)
- [WebSphere Application Server](#) от [IBM](#) (основан на [Apache](#) и платформе [J2EE](#))
- [Zend Framework](#) — от [Zend Technologies](#)
- [Zope](#) является [объектно-ориентированным сервером приложений](#) написанным на [Python](#)

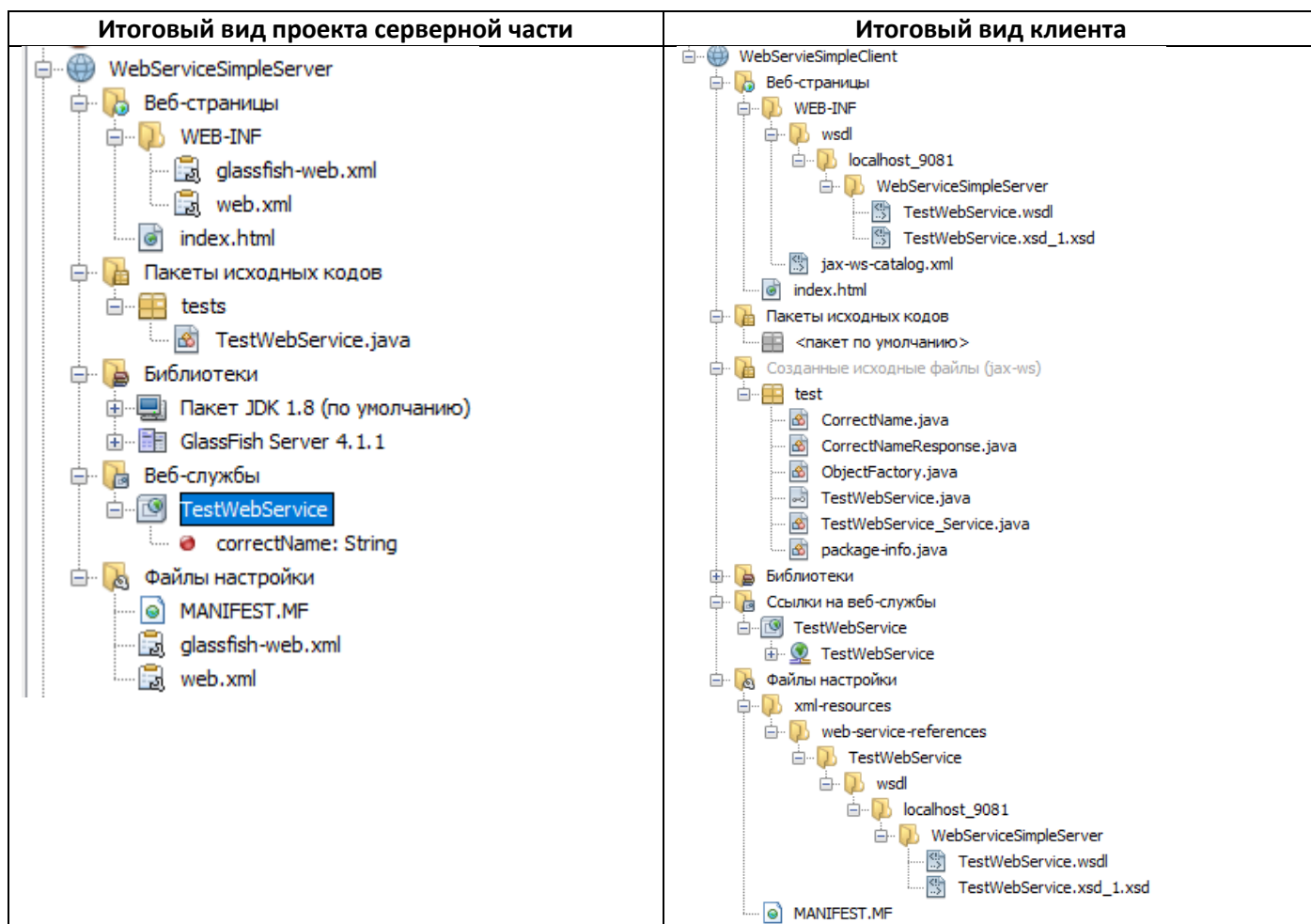
При создании Вэб-служб используют два подхода (в основном одновременно): создание при использовании SOAP протоколов (так называемые «Big Web service») и создание на основе архитектурных принципов REST (так называемые RESTfull сервисы).

Big Web service основаны на множестве протоколов и спецификаций сложнее в разработке, но поддерживают транзакции и уровни безопасности

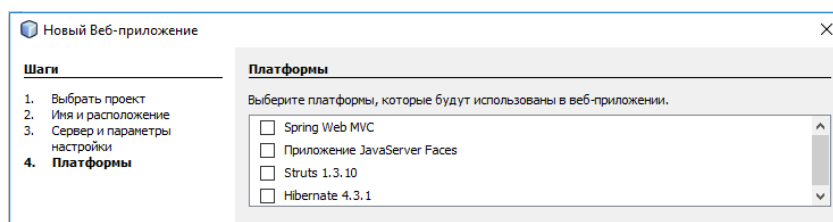
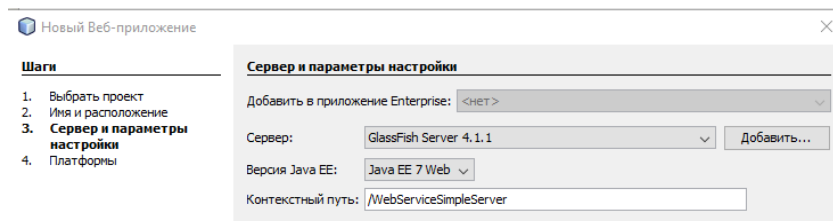
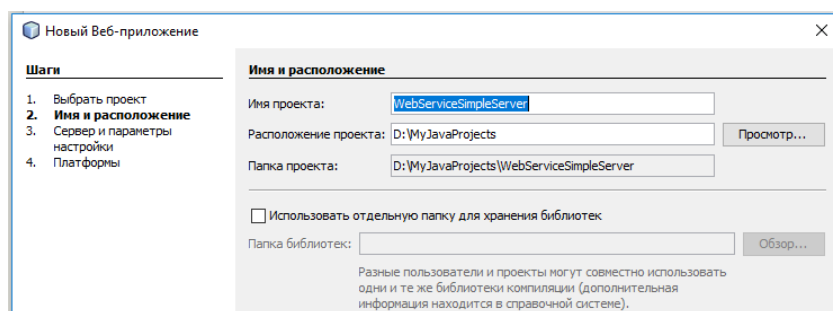
RESTfull сервисы гораздо быстрее работают и намного легче в разработке, основаны на HTTP запросах и наследуют все особенности безопасности таковых.

## Примеры использования вэб-сервисов

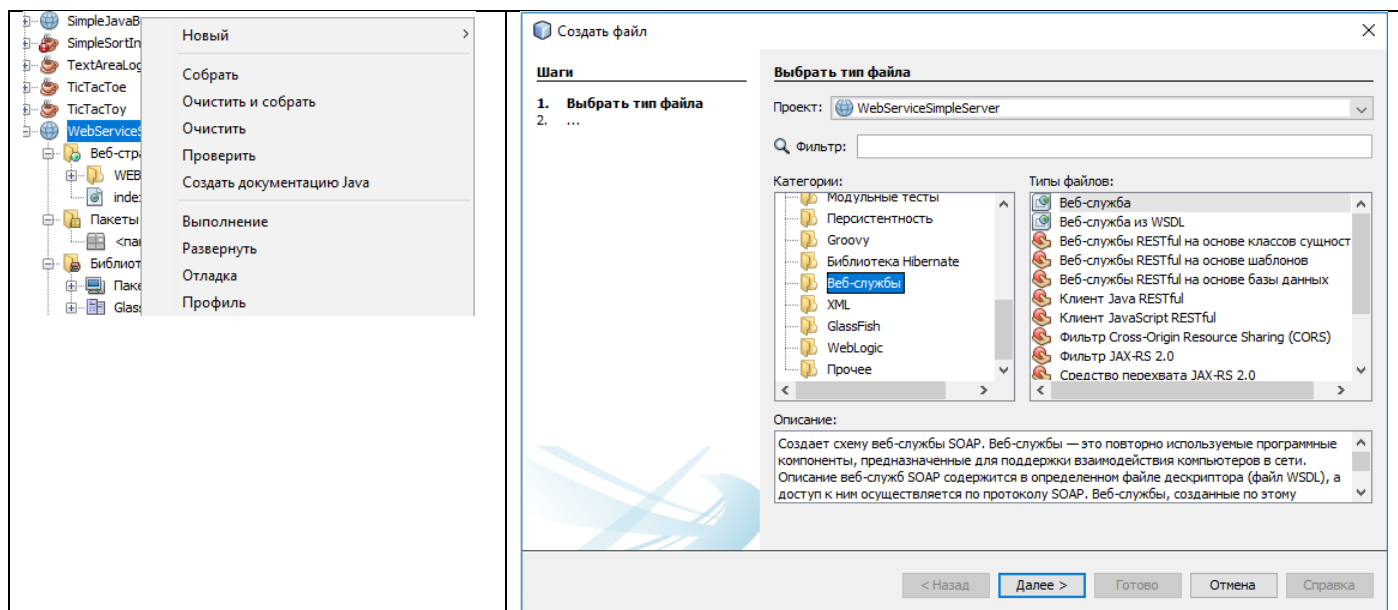
### Пример 1 Простая вэб-служба



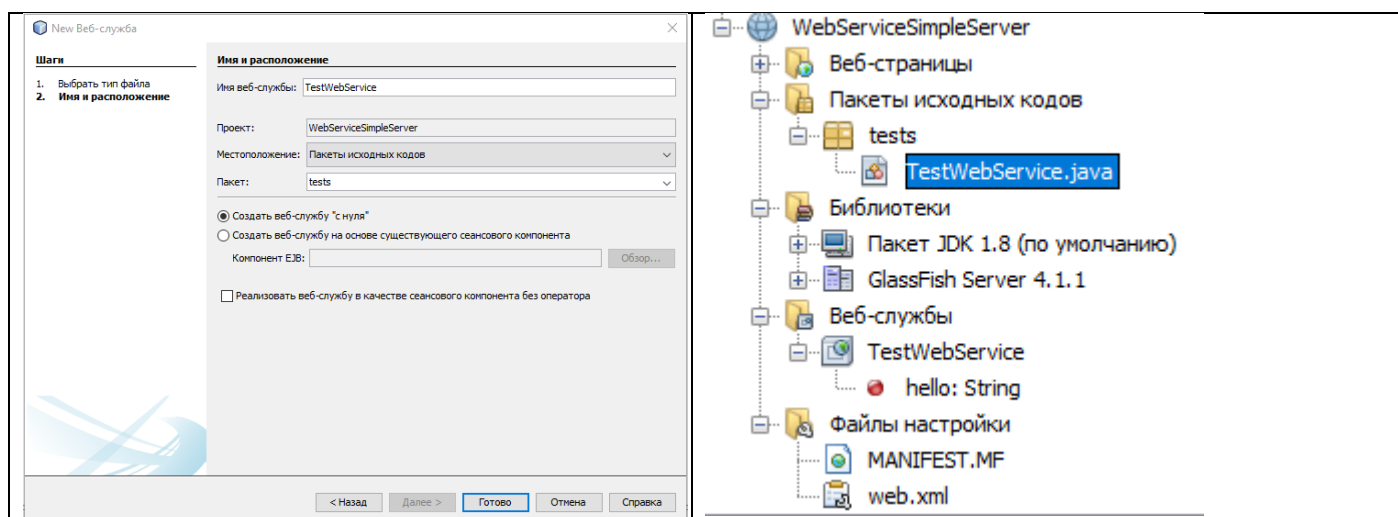
Создаем серверную часть вэб-службы. Создаем проект Java Web обычное Java web-application



Далее добавляем в него сервис: 1) через контекстное меню, которое появляется при вызове на имени проекта  
2) добавляем в проект веб-службу (New->Other)



После выполнения последнего шага получим :



### Исходный текст TestWebService.java

```
package tests;

import javax.xml.ws.WebService;
import javax.xml.ws.WebMethod;
import javax.xml.ws.WebParam;

@WebService(serviceName = "TestWebService")
public class TestWebService {

    /**
     * This is a sample web service operation
     */
    @WebMethod(operationName = "hello")
    public String hello(@WebParam(name = "name") String txt) {
        return "Hello " + txt + " !";
    }
}
```

### В результате файл TestWebService.java должен выглядеть так

```
package tests;

import javax.xml.ws.WebService;
import javax.xml.ws.WebMethod;
import javax.xml.ws.WebParam;

@WebService(serviceName = "TestWebService")
public class TestWebService {

    /**
     * This is a sample web service operation
     */
    @WebMethod(operationName = "correctName")
    public String correctName(@WebParam(name = "name") String name) {
        return "My Name is " + name;
    }
}
```

Протестируем нашей службы – развернем проект (Deploy) - и зайдем после этого в контекстное меню

Выход X

WebServiceSimpleServer (run-deploy) x

Процесс базы данных Java DB x

GlassFish Server 4.1.1 x

init:

deps-module-jar:

deps-ear-jar:

deps-jar:

library-inclusion-in-archive:

library-inclusion-in-manifest:

compile:

compile-jsp:

Запуск GlassFish Server 4.1.1

GlassFish Server 4.1.1 выполняется.

Отмена развертывания...

Развертывание на месте на D:\MyJavaProjects\WebServiceSimpleServer\build\we

run-deploy:

СБОРКА УСПЕШНО ЗАВЕРШЕНА (общее время: 44 секунды)

Выход X

WebServiceSimpleServer (run-deploy) x

Процесс базы данных Java DB x

GlassFish Server 4.1.1 x

Warning: Instance could not be initialized. Class=interface org.glassfish.grizzly.http.ser

Info: Created HTTP listener http-listener-2 on host/port 0.0.0.0:8181

Info: Grizzly Framework 2.3.23 started in: 16ms - bound to [/0.0.0.0:8181]

Warning: Instance could not be initialized. Class=interface org.glassfish.grizzly.http.ser

Info: Created HTTP listener http-listener-1 on host/port 0.0.0.0:9081

Info: Grizzly Framework 2.3.23 started in: 15ms - bound to [/0.0.0.0:9081]

Info: Registered com.sun.enterprise.glassfish.bootstrap.osgi.EmbeddedOSGiGlassFishImpl#95f:

Info: visiting unvisited references

Info: visiting unvisited references

Info: WebService Endpoint deployed TestWebService

listening at address at http://DESKTOP-V7KQ1U2:9081/WebServiceSimpleServer/TestWebService.

Info: Loading application [WebServiceSimpleServer] at [/WebServiceSimpleServer]

Info: WebServiceSimpleServer was successfully deployed in 330 milliseconds.

Веб-службы

TestWebS

correct

Файлы настр

MANIFEST

glassfish-v

glassfish-web.xml - На

version="1.0" enco

PUBLIC "-//GlassFis

glassfish-web-a

Открыть

Обновить...

Добавить операцию...

Тестировать веб-службу

Правка атрибутов веб-службы...

Настройка обработчиков...

Создать и копировать файл WSDL

Создать обертку SOAP по HTTP

Удалить

http://localhost:9081/WebServiceSimpleServer/TestWebService?Tester

TestWebService Web Service Tester

This form will allow you to test your web service implementation ([WSDL File](#))

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name

Methods :

public abstract java.lang.String tests.TestWebService.correctName(java.lang.String)

correctName (Бася)

Должны увидеть нечто подобное:

http://localhost:9081/WebServiceSimpleServer/TestWebService?Tester

correctName Method invocation

Method parameter(s)

Type	Value
java.lang.String	Бася

Method returned

java.lang.String : "My Name is Бася"

SOAP Request

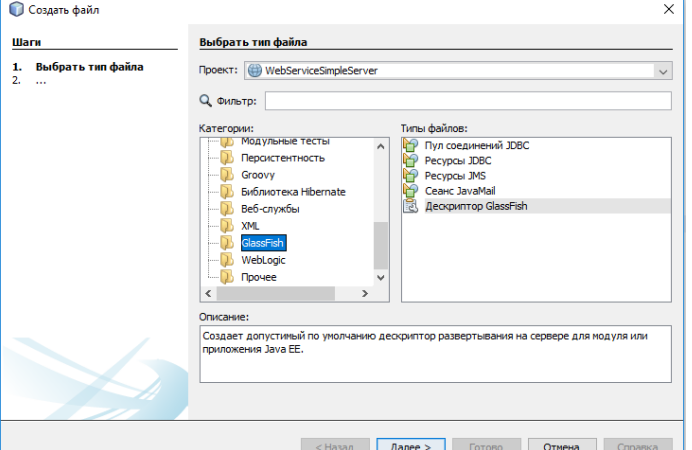
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"><SOAP-ENV:Header/><S:Body><ns2:correctName xmlns:ns2="http://tests/"><name>Бася</name></ns2:correctName></S:Body></S:Envelope>

SOAP Response

<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"><SOAP-ENV:Header/><S:Body><ns2:correctNameResponse xmlns:ns2="http://tests/"><return>My Name is Бася</return></ns2:correctNameResponse></S:Body></S:Envelope>

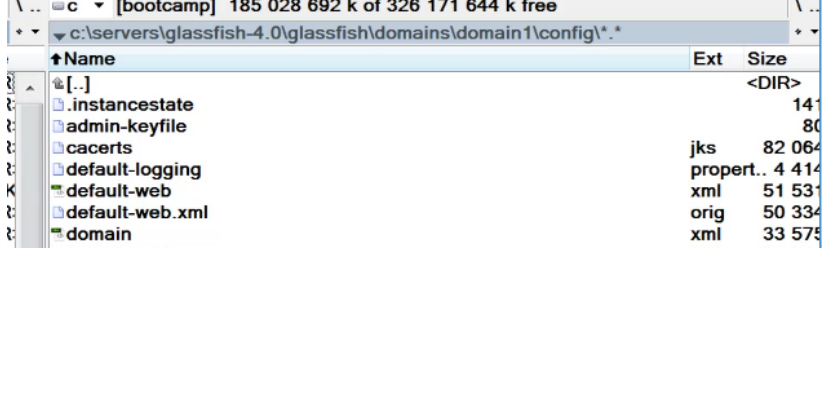
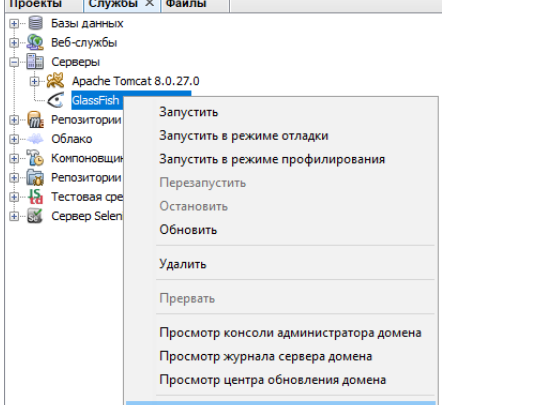
**Особенности локализации Glassfish сервера**

1) добавляем файл настройки сервера glassfish-web.xml в проект (это файл будет содержать настройки для этого проекта и создается из контекстного меню проекта)

Конечный вариант содержимого файла glassfish-web.xml	Внешний вид контекстного меню NEW->OTHER
<pre>&lt;?xml version="1.0" encoding="UTF-8"?&gt; &lt;!DOCTYPE glassfish-web-app PUBLIC "-//GlassFish.org//DTD GlassFish Application Server 3.1 Servlet 3.0//EN" "http://glassfish.org/dtds/glassfish-web-app_3_0-1.dtd"&gt; &lt;glassfish-web-app error-url=""&gt;   &lt;class-loader delegate="true"/&gt;   &lt;jsp-config&gt;     &lt;property name="keepgenerated" value="true"&gt;       &lt;description&gt;Keep a copy of the generated servlet class' java code.&lt;/description&gt;     &lt;/property&gt;   &lt;/jsp-config&gt;   &lt;parameter-encoding default-charset="UTF-8" /&gt; &lt;/glassfish-web-app&gt;</pre>	

2) корректируем настройки виртуальной машины самой java

Вносим изменения в файл domain.xml путь к которому можно узнать в настройках сервера:

	
--	---

