class: inverse, center, middle

.training360-logo[Training360]

# JAVAX-WEB - Webszolgáltatások megvalósítása Java platformon

## Tematika

* Webszolgáltatások SAAJ API használatával
* XML és Java objektumok közötti megfeleltetés Java API for XML Binding (JAXB) 2.0 használatával
* SOAP alapú webszolgáltatások fejlesztése Java API for XML Web Services (JAX-WS) 2.0 használatával
* REST alapú webszolgáltatások Java API for RESTful Web Services (JAX-RS) 2.0 használatával

## Források

* Martin Kalin: Java Web Services Up and Running (O'Reilly)
* The Java EE 5 - 6 Tutorial (Addison Wesley)
* Bill Burke: RESTful Java with JAX-RS (O'Reilly)
* Mark D. Hansen: SOA Using Java Web Services (Prentice Hall)

## Opcionális források

* Steve Graham, Doug Davis, ...: Building Web Services with Java: Making Sense of XML, SOAP, WSDL, and UDDI, 2nd Edition (Developer’s Library)
* Antonio Goncalves: Beginning Java EE 7 (Apress)
* Arun Gupta: Java EE 7 Essentials (O'Reilly)

## Webszolgáltatás

* W3C definíció: hálózaton keresztüli gép-gép együttműködést támogató szoftverrendszer
* Platform független
* Szereplők
  + Szolgáltatást nyújtó
  + Szolgáltatást használni kívánó
  + Registry

## SOAP alapú webszolgáltatások

* Szabványokon alapuló
  + SOAP
  + WSDL
  + UDDI
* Kapcsolódó szabványok
  + HTTP
  + XML
  + XSD
  + Namespaces
  + (XSLT)

## Kapcsolódó szabványok, ajánlások

* WS-I
  + Open industry consortium
  + Profilok: specifikált verziók (SOAP, WSDL, stb.) + megszorítások az interoperabilitás elősegítésére
  + Basic Profile
* WS-\*
  + WS-Security
  + WS-Reliability
  + WS-Transaction

## SOA

* Service Oriented Architecture
* Újrafelhasználható szolgáltatások
* Szolgáltatásokat igénybe vevők
* Service contract, interfész ismert, implementáció rejtett
* Enterprise Service Bus (ESB)
* Üzleti folyamatok, orchestration, BPEL
* Rule engine
* Üzenet orientált
* Lazán kapcsolt
* Tervezési minták
* Governance, Management, Security

## SOAP

* W3C által karbantartott 1.1, 1.2
* Kiterjeszthető

## SOAP üzenet

## SOAP

* Stílus
  + RPC-stílus
  + Document stílus
* Kódolás
  + Literal-kódolás
  + SOAP-kódolás = encoded
* Paraméter
  + Unwrapped (bare)
  + Wrapped

## SOAP ajánlás

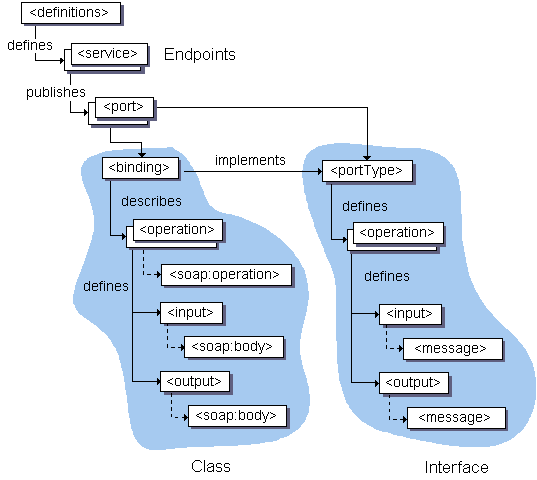
* Leggyakrabban használt
  + Document/literal

http://www.ibm.com/developerworks/webservices/library/ws-whichwsdl/

## WSDL

* Web Services Description Language
* WSDL dokumentum: több állományból

## WSDL felépítése



WSDL

## Implementációs megközelítések

* Megközelítések
  + Bottom up: kód alapján
  + Top down: WSDL alapján
  + Meet in the middle
* Üzleti logika != interfész
  + Két szolgáltatás, két osztály
  + Megoldás: wrapper réteg

## Webszolgáltatás stack

* JSR 67 - SAAJ
* JSR 222 – JAXB 2.0
* JSR 224 – JAX-WS 2.0
* JSR 181 – Web Services Metadata 2.0
* JSR 109 – WS EE
* Java centric, start with Java approach
* Toolset, not framework (nem ad egy egységes környezetet, absztrakciót)

## Stack felépítése

* Invocation
* Serialization
* Deployment

## Invocation

* Transport
* Handlers
* Dispatching (mely Java komponenst kell meghívni)
* SOAP fault

## Serialization

* Java példány XML dokumentummá alakítása
* Különbség: mapping, binding
  + Egy osztály, több mapping
  + Valódi mapping framework: Castor

## Deployment

* Java target
* Mapping WSDL operations to Java target
* Serialization context (Java objektumok és XML dokumentumok közötti megfeleltetéséért)
* Publishing WSDL
* SOAP handlers
* Endpoint listener (URL-en figyelés)

## JAX-WS

* JAX-RPC 1.1 továbbfejlesztése
* RPC stílus és üzenet alapú
* Cél: fejlesztés egyszerűsítése
  + Alapértelmezett értékek, annotációk
  + Standard WSDL/Java mapping
  + Standard serialization context (delegálja a JAXB-nak)
* Java SE-ben is használható
* Java EE-ben
  + Web alkalmazásban: POJO (nem kell EJB konténer)
  + EJB: Stateless session bean (biztonság, tranzakció-kezelés)

## Implementációk

* JAX-WS Reference Implementation
  + https://javaee.github.io/metro-jax-ws/
  + Legfrissebb verzió: JAX-WS RI 2.3.0
* Apache Axis2
  + http://axis.apache.org/axis2/java/core/
* CXF
  + http://cxf.apache.org/

## Webszolgáltatás stack verziók

* Java SE 8 - JAX-WS 2.2

$ java -version  
java version "1.8.0\_101"  
Java(TM) SE Runtime Environment (build 1.8.0\_101-b13)  
Java HotSpot(TM) 64-Bit Server VM (build 25.101-b13, mixed mode)  
$ wsimport -version  
wsimport version "2.2.9"  
$ wsimport -fullversion  
wsimport full version "JAX-WS RI 2.2.9-b130926.1035  
 svn-revision#8c29a9a53251ff741fca1664a8221dc876b2eac8"  
$ wsgen -version  
wsgen version "2.2.9"  
$ wsgen -fullversion  
wsgen full version "JAX-WS RI 2.2.9-b130926.1035  
 svn-revision#8c29a9a53251ff741fca1664a8221dc876b2eac8"

## Webszolgáltatás stack verziók EE környezetben

* Java EE 6, Java EE 7 - JAX-WS 2.2

## Service endpoint

* Java osztály (@WebService annotációval ellátva)
* Explicit SEI (Service Endpoint Interface)
  + endpointInterface attribútummal megadva, implementálva
  + Ennek metódusai kerülnek publikálásra
* Implicit SEI
  + Minden publikus metódus publikálásra kerül

## Szerver implementálása

* @WebService
  + serviceName, portName
  + targetNamespace
  + wsdlLocation
* @WebMethod - összes publikus publikálásra kerül, ezzel csak konfigurálni
  + operationName
  + action
  + exclude: nem lesz benne a WSDL-ben

## Binding

* @SOAPBinding
  + style: DOCUMENT / RPC
  + use: LITERAL / ENCODED

## Paraméterek és visszatérési értékek

* @WebResult
  + name
  + targetNamespace
* @WebParam
  + name
  + targetNamespace
  + mode: IN / OUT / INOUT
* @RequestWrapper
* @ResponseWrapper
* @OneWay

## Kliens implementálása

* Java API
* Generált osztályok

Parancssorból

wsimport -d <directory> -p <pkg> -s <directory> -target <version> -keep wsdl.wsdl

JAX-WS Maven Plugin, wsimport goal

## Kivételkezelés

* <Fault> tag
* Hibakezelés
  + RuntimeException leszármazott (unchecked)
  + Egyéb Exception leszármazott (checked), @WebFault annotáció
  + SOAPFaultException

SOAPFactory soapFactory = SOAPFactory.newInstance();  
SOAPFault soapFault = soapFactory.createFault(  
 "Cannot div parameter by 2.",  
 new QName("http://schemas.xmlsoap.org/soap/envelope/", "Client"));  
throw new SOAPFaultException(soapFault);

## WebServiceContext

* DI (@Resource annotációval)
* Hozzáférés pl. a HTTP stackhez, pl. header

@Resource  
private WebServiceContext webServiceContext;

Map<String, List<String>> headers = (Map<String, List<String>>) webServiceContext  
 .getMessageContext()  
 .get(MessageContext.HTTP\_REQUEST\_HEADERS);

## Request context

* A port BindingProvider példány

BindingProvider bindingProvider = (BindingProvider) echo;  
Map<String, List<String>> headers = new HashMap<>();  
headers.put("foo", Arrays.asList("bar"));  
bindingProvider.getRequestContext().put(MessageContext.HTTP\_REQUEST\_HEADERS, headers);

## Validáció

* @SchemaValidation annotáció a com.sun.xml.ws.developer csomagban

## Tesztelhetőség – publikálás

* Java 6 API – Endpoint.publish
* Embedded web container
* Standalone container

## Tesztelhetőség - hívás

* Kliens
* SAAJ/SOAPMessage
* HTTP/XML (Parser, XPath, XMLUnit)

## JAX-WS 2.0

* Java/WSDL mapping
  + Java -> WSDL
  + WSDL -> Java
  + WSDL -> Java -> WSDL
  + Annotációk
  + Embedded binding declarations
  + External binding declarations
* Static WSDL

## JAX-WS 2.0

* Dinamic/static clients
* Java interface proxy client
* XML invocation (Dispatcher, Provider)

## JAX-WS 2.0

* Handler @HandlerChain
  + Logical level (Source, JAXB)
  + Protocol level (SOAP, SAAJ)

## JAX-WS 2.0

* SOAP Binding
  + mustUnderstand
  + next
  + ultimateReceiver
  + Header
  + SOAP fault
  + MTOM, SwA

## JAX-WS 2.0

* HTTP binding
  + REST
* Exception to SOAP fault
* Asynch call
  + Polling
  + Callback
* Oneway
* Client side thread handling
  + Thread pool

## JAX-WS 2.0

* WSDL styles
  + Unwrapped, wrapped
  + Rpc/literal = RPC
  + Document/literal wrapped = Document Wrapped
* XML catalog
  + OASIS XML catalog
  + Schema location feloldás

## JAX-WS 2.0

* Holder
* Java SE: runtime endpoint publishing

## WS-Metadata 2.0

* Easy of use
* WSDL mapping annotation
  + @Webservice, @WebMethod, @WebResult, @WebParam
* WSDL binding annotations
  + @SOAPBinding
* Handler
  + @HandlerChain
* SIB
* Automatic deployment

## WSEE 1.2

* Kötelező
  + SIB
* Opcionális
  + SEI
  + WSDL
  + webservices.xml

## WSEE 1.2

* Servlet/EJB endpoint
* DI
  + @WebServiceRef
  + @Resource
* Interceptor
  + AOP

## RESTful webszolgáltatások

* Roy Fielding: Architectural Styles and the Design of Network-based Software Architectures, 2000
* Client-server
* Representational state transfer
* Létező technológiák: URI, HTTP, XML, JSON
* Web Application Description Language (WADL) – nem elterjedt
* AJAX világ segítette az elterjedését

## REST fogalmak

* Egyedileg címezhető erőforrások: resource, URI
* Uniform, constrained interface for manipulate resources
* Representation-oriented, content negotiation
* Stateless
* Hypermedia As The Engine Of Application State (HATEOAS): embedded links

## REST tulajdonságok

* MIME: type/subtype;name=value;name=value...
* Stateless: helyette url paraméterek, cookie, hidden field
* Cacheable
* Layered

## JAX-RS

* Java EE része
* Root resource classes: POJO
* Annotációk, akár interfészen, akár absztrakt ősosztályon
* Resource methods: URI & operation to method mapping
* Request method designator
* Parameter injection
* BodyReader, BodyWriter
* Exception mappers
* Content negotiation

## Verziók

* JAX-RS 1.0, 1.1 - JSR 311
* JAX-RS 2.0 - JSR 339
* Java EE 6 - JAX-RS 1.1
* Java EE 7 - JAX-RS 2

## JAX-RS annotációk

* @Path
  + Illesztés, template, regexp, precedencia
  + Subresource
* @Produces
* @Consumes
* @GET, @POST, @PUT, @DELETE, @HEAD és @OPTIONS, @HttpMethod
* Response, Response.Status
* WebApplicationException
* StreamingOutput

## JAX-RS deployment

* web.xml
* Application
  + Per request
  + Singleton

## Injection

* @PathParam, @QueryParam, @MatrixParam, @CookieParam, @HeaderParam és @FormParam
* Per request resource esetén attribútumra is
* @PathParam List<PathSegment> paraméter
* Cookie, @CookieParam annotációval
* @DefaultValue annotáció
* @Encoded annotáció

## Provider

* MessageBodyReader, Writer
* ContextResolver
* ExceptionMapper

## @Context annotáció

* A következőkbe injektálhatók
  + Resource
  + Provider
  + Application leszármazott

## @Context annotáció

* A következők injektálhatók
  + Application (önmagába nem)
  + UriInfo
  + HttpHeaders
  + Request
  + SecurityContext
  + Providers

## JAX-RS content handler

* MessageBodyReader, Writer
* StreamingOutput
* InputStream, Reader
* File
* byte[]
* String, char[] - @Produces charset
* MultivaluedMap<String, String>
* Source

## JAXB

* @XmlRootElement, @XmlType
* JAXBElement
* application/xml, text/xml, vagy application/\*+xml
* ContextResolver JAXBContext példányosításához, paraméterezéséhez

## JAX-RS 2.0 újdonságok

* Client API
* Async
* HATEOAS (Hypermedia)
* Annotations
* Validation
* Filters and Handlers
* Content negotiation

## SOAPUi

* SOA teszt eszköz
* Ingyenes és kereskedelmi verzió
* Webszolgáltatások felderítése, hívása,
* Szimulálás és mock
* Funkcionális és terheléses tesztelés
* IDE-plugins
* SOAPUi – nightly build – jó support

## SOAPUi

* Új SOAP projekt WSDL alapján
* Új REST projekt
* Request létrehozása
* Request futtatása
* Assert létrehozása
* Mock service létrehozása, customresponses
* Mock service futtatása

## SOAPUi haladó eszközök

* Groovy scripting
* Naplóelemzés
* WS-SecuritySupport
* Futtatás parancssorból
* Futtatás Mavennel, maven-soapui-plugin segítségével

## Teljesítmény - XML

* Factory reuse
* Parser pool
* JAXBContext reuse
* Memóriaigény
* Validation
* Schema cache
* DTD – custom entity resolver
* Partial processing
* Selectiong right API: DOM, SAX, StaX, JAXB

## Teljesítmény - WS

* Message size
* Schema
* Endpoint implementation
* Handlers
* Fast infoset
* HTTP compression
* Reuse client proxy