

[La meva pàgina inicial](#) / [Els meus cursos](#) / [DAW M07B2 Desenvolupament web en entorn servidor \(Bloc 2\)](#)  
/ [Setmana 7. Serveis web SOAP amb Spring Web Services.](#) / [Qüestionari EAC6](#)

<b>Començat el</b>	dilluns, 22 novembre 2021, 16:45
<b>Estat</b>	Acabat
<b>Completat el</b>	dimarts, 30 novembre 2021, 11:37
<b>Temps emprat</b>	7 dies 18 hores
<b>Qualificació</b>	9,50 sobre 10,00 (95%)

# Mòdul 7 B2 – Desenvolupament web en entorn servidor

## UF4 – Serveis web. Pàgines dinàmiques interactives. Webs Híbrids

### Unitat 6 – Serveis web amb Spring

EAC6

(Curs 2021–22 / 1r semestre)

#### Presentació i resultats d'aprenentatge

Aquest exercici d'avaluació continuada (EAC) es correspon amb els continguts treballats a la unitat 6 “**Serveis web amb Spring**”

Els **resultats d'aprenentatge** que es plantegen són:

- Desenvolupa serveis web analitzant el seu funcionament i implantant l'estructura dels seus components.

#### Criteris d'avaluació

La puntuació màxima assignada a cada activitat s'indica a l'enunciat.

Els criteris que es tindran en compte per avaluar el treball de l'alumnat són els següents:

- Identifica les característiques pròpies i l'àmbit d'aplicació dels serveis web.
- Reconeix els avantatges d'utilitzar serveis web per proporcionar accés a funcionalitats incorporades a la lògica de negoci d'una aplicació.
- Determina les tecnologies i els protocols implicats en la publicació i utilització de serveis web.
- Programa un servei web.
- Crea el document de descripció del servei web.
- Verifica el funcionament del servei web.
- Consumeix el servei web.

#### Forma de lliurament

Us demanem que, a part de contestar aquest qüestionari, copieu totes les preguntes i respostes en un document \*.odt.

En aquest document hi heu d'afegir totes les preguntes i respostes de TOTS els qüestionaris de l'EAC en curs i, finalment, entregar-ho a la tasca EACx corresponent.

(si copieu les respostes al document sempre tindreu una còpia en cas que s'acabi la sessió del navegador també. ;D)

---

Tenim una botiga de cotxes que disposa d'una web de catàleg. Volem que la web actuï com a servei de dades per a una app que estem desenvolupant i hem pensat fer un webservice per a resoldre aquest problema.

La nostra web ens ha de permetre consultar el catàleg complet dels cotxes disponibles (ha de mostrar model i marca. Pot ser en un mateix string), consultar aquesta mateixa informació, afegint l'estoc actual i el preu en una petició de detall (utilitzant el model del cotxe com a paràmetre) i consultar el llistat de cotxes del mateix fabricant (passant per paràmetre el fabricant).

Escriuiu el wsdl que defineix el contracte del web service. Com sempre, podeu contestar amb el codi o pujar-ne el fitxer

//cotxe.xsd

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://cat.xtec.ioc/domain/cotxe" targetNamespace="http://cat.xtec.ioc/domain/cotxe" elementFormDefault="qualified"> <xs:complexType name="cotxe"> <xs:sequence> <xs:element name="model" type="xs:string"/> <xs:element name="marca" type="xs:string"/> <xs:element name="estoc" type="xs:int"/> <xs:element name="preu" type="xs:float"/> </xs:sequence> </xs:complexType> </xs:schema>
```

//cotxeoperations.xsd

**//getCatalagCotxesDetailByModelResponse:** Ens retorna un estring amb la informació específica del catàleg de cotxes però únicament el del model que especifiquem en el paràmetre

**//getCatalagCotxesByModelResponse(tot i que no es demana,però en un principi no entenia el que demanava l'enunciat):** Ensretorna un objecte de tipus Cotxe, però únicament el del model que especifiquem en el paràmetre.

**//getLlistatCotxesByMarcaResponse:** Ens retorna un llistat de tots el cotxes que coincideixen amb la marca, que es passa per paràmetre.

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://cat.xtec.ioc/domain/cotxe/services"
    targetNamespace="http://cat.xtec.ioc/domain/cotxe/services" xmlns:cotxe="http://cat.xtec.ioc/domain/cotxe" elementFormDefault="qualified">

    <xs:import namespace="http://cat.xtec.ioc/domain/cotxe" schemaLocation="cotxe.xsd"/>

    <xs:element name="getCatalagCotxesDetallByModelRequest">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="model" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="getCatalagCotxesDetallByModelResponse">
        <xs:complexType>
            <xs:sequence>
                <xs:element maxOccurs="unbounded" name="catalagCotxesDetallByModel" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:element name="getCatalagCotxesByModelRequest">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="model" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="getCatalagCotxesByModelResponse">
        <xs:complexType>
            <xs:sequence>
                <xs:element maxOccurs="unbounded" name="catalagCotxesByModel" type="cotxe:cotxe"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:element name="getLlistatCotxesByMarcaRequest">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="marca" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="getLlistatCotxesByMarcaResponse">
        <xs:complexType>
            <xs:sequence>
                <xs:element maxOccurs="unbounded" name="llistatCotxesByMarca" type="cotxe:cotxe"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

</xs:schema>
```

## //cotxeoperations.wsdl

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:sch="http://cat.xtec.ioc/domain/cotxe/services" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://cat.xtec.ioc/domain/cotxe/services" targetNamespace="http://cat.xtec.ioc/domain/cotxe/services">
  <wsdl:types>
    <xs:schema xmlns:cotxe="http://cat.xtec.ioc/domain/cotxe" xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" targetNamespace="http://cat.xtec.ioc/domain/cotxe/services">
      <xs:import namespace="http://cat.xtec.ioc/domain/cotxe" schemaLocation="cotxe.xsd"/>
      <xs:element name="getCatalagCotxesDetallByModelRequest">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="model" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="getCatalagCotxesDetallByModelResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element maxOccurs="unbounded" name="catalagCotxesDetallByModel" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="getCatalagCotxesByModelRequest">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="model" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="getCatalagCotxesByModelResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element maxOccurs="unbounded" name="catalagCotxesByModel" type="cotxe:cotxe"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
```

```
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="getListatCotxesByMarcaRequest">
<xs:complexType>
<xs:sequence>
<xs:element name="marca" type="xs:string"/>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="getListatCotxesByMarcaResponse">
<xs:complexType>
<xs:sequence>
<xs:element maxOccurs="unbounded" name="llistatCotxesByMarca" type="cotxe:cotxe"/>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>
</wsdl:types>
<wsdl:message name="getListatCotxesByMarcaRequest">
<wsdl:part element="tns:getListatCotxesByMarcaRequest" name="getListatCotxesByMarcaRequest"> </wsdl:part>
</wsdl:message>
<wsdl:message name="getCatalagCotxesByModelResponse">
<wsdl:part element="tns:getCatalagCotxesByModelResponse" name="getCatalagCotxesByModelResponse"> </wsdl:part>
</wsdl:message>
<wsdl:message name="getCatalagCotxesDetallByModelResponse">
<wsdl:part element="tns:getCatalagCotxesDetallByModelResponse" name="getCatalagCotxesDetallByModelResponse"> </wsdl:part>
</wsdl:message>
<wsdl:message name="getCatalagCotxesDetallByModelRequest">
<wsdl:part element="tns:getCatalagCotxesDetallByModelRequest" name="getCatalagCotxesDetallByModelRequest"> </wsdl:part>
</wsdl:message>
<wsdl:message name="getCatalagCotxesByModelRequest">
<wsdl:part element="tns:getCatalagCotxesByModelRequest" name="getCatalagCotxesByModelRequest"> </wsdl:part>
</wsdl:message>
<wsdl:message name="getListatCotxesByMarcaResponse">
<wsdl:part element="tns:getListatCotxesByMarcaResponse" name="getListatCotxesByMarcaResponse"> </wsdl:part>
</wsdl:message>
<wsdl:portType name="CotxesPort">
<wsdl:operation name="getListatCotxesByMarca">
<wsdl:input message="tns:getListatCotxesByMarcaRequest" name="getListatCotxesByMarcaRequest"> </wsdl:input>
<wsdl:output message="tns:getListatCotxesByMarcaResponse" name="getListatCotxesByMarcaResponse"> </wsdl:output>
</wsdl:operation>
<wsdl:operation name="getCatalagCotxesByModel">
<wsdl:input message="tns:getCatalagCotxesByModelRequest" name="getCatalagCotxesByModelRequest"> </wsdl:input>
<wsdl:output message="tns:getCatalagCotxesByModelResponse" name="getCatalagCotxesByModelResponse"> </wsdl:output>
</wsdl:operation>
<wsdl:operation name="getCatalagCotxesDetallByModel">
<wsdl:input message="tns:getCatalagCotxesDetallByModelRequest" name="getCatalagCotxesDetallByModelRequest"> </wsdl:input>
<wsdl:output message="tns:getCatalagCotxesDetallByModelResponse" name="getCatalagCotxesDetallByModelResponse"> </wsdl:output>
</wsdl:operation>
</wsdl:portType>
<wsdl:binding name="CotxesPortSoap11" type="tns:CotxesPort">
<soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
<wsdl:operation name="getListatCotxesByMarca">
<soap:operation soapAction=""/>
<wsdl:input name="getListatCotxesByMarcaRequest">
<soap:body use="literal"/>
</wsdl:input>
<wsdl:output name="getListatCotxesByMarcaResponse">
<soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="getCatalagCotxesByModel">
<soap:operation soapAction=""/>
<wsdl:input name="getCatalagCotxesByModelRequest">
<soap:body use="literal"/>
</wsdl:input>
<wsdl:output name="getCatalagCotxesByModelResponse">
<soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="getCatalagCotxesDetallByModel">
<soap:operation soapAction=""/>
<wsdl:input name="getCatalagCotxesDetallByModelRequest">
<soap:body use="literal"/>
</wsdl:input>
<wsdl:output name="getCatalagCotxesDetallByModelResponse">
<soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="CotxesPortService">
<wsdl:port binding="tns:CotxesPortSoap11" name="CotxesPortSoap11">
<soap:address location="http://localhost:8080/soapws"/>
</wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

## //Missatge SOAP getCatalagCotxesDetallByModelRequest:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ser="http://cat.xtec.ioc/domain/cotxe/services"> <soapenv:Header/> <soapenv:Body>
<ser:getCatalagCotxesDetallByModelRequest> <ser:model>Micra</ser:model> </ser:getCatalagCotxesDetallByModelRequest> </soapenv:Body> </soapenv:Envelope>
```

## //Resposta getCatalagCotxesDetallByModelResponse:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"> <SOAP-ENV:Header/> <SOAP-ENV:Body> <ns2:getCatalagCotxesDetallByModelResponse
xmlns:ns2="http://cat.xtec.ioc/domain/cotxe/services"> <ns2:catalagCotxesDetallByModel>CATÀLAG DE COTXES El Model és: Micra La marca és: nissan L'estoc és: 500 El preu és
25000.0</ns2:catalagCotxesDetallByModel> </ns2:getCatalagCotxesDetallByModelResponse> </SOAP-ENV:Body> </SOAP-ENV:Envelope>
```

//Missatge SOAP getCatalogCotxesByModelRequest:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ser="http://cat.xtec.ioc/domain/cotxe/services">
```

```
<soapenv:Header/> <soapenv:Body> <ser:getCatalogCotxesByModelRequest> <ser:model>Micra</ser:model>
</ser:getCatalogCotxesByModelRequest> </soapenv:Body> </soapenv:Envelope>
```

Resposta getCatalogCotxesByModelResponse

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns3:getCatalogCotxesByModelResponse xmlns:ns2="http://cat.xtec.ioc/domain/cotxe" xmlns:ns3="http://cat.xtec.ioc/domain/cotxe/services">
      <ns3:catalogCotxesByModel>
        <ns2:model>Micra</ns2:model>
        <ns2:marca>nissan</ns2:marca>
        <ns2:estoc>500</ns2:estoc>
        <ns2:preu>25000.0</ns2:preu>
      </ns3:catalogCotxesByModel>
    </ns3:getCatalogCotxesByModelResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

//InMemoryCotxeRepository

```

package cat.xtec.ioc.domain.repository.impl;
import ioc.xtec.cat.domain.cotxe.Cotxe;
import java.util.ArrayList;
import java.util.List;
import javax.annotation.PostConstruct;
import org.springframework.stereotype.Repository;
import org.springframework.util.Assert;
@Repository
public class InMemoryCotxeRepository {
    private static final List<Cotxe> cotxes = new ArrayList<Cotxe>();
    @PostConstruct
    public void initData() {
        Cotxe nissan = new Cotxe();
        nissan.setMarca("nissan");
        nissan.setModel("LEAF");
        nissan.setEstoc(50);
        nissan.setPreu(20000);
        cotxes.add(nissan);

        Cotxe nissan2 = new Cotxe();
        nissan2.setMarca("nissan");
        nissan2.setModel("Micra");
        nissan2.setEstoc(500);
        nissan2.setPreu(25000);
        cotxes.add(nissan2);

        Cotxe tesla = new Cotxe();
        tesla.setMarca("Tesla");
        tesla.setModel("Model S");
        tesla.setEstoc(200);
        tesla.setPreu(75000);
        cotxes.add(tesla);
        Cotxe ford = new Cotxe();
        ford.setMarca("Ford");
        ford.setModel("B-Max");
        ford.setEstoc(300);
        ford.setPreu(30000);
        cotxes.add(ford);
        for (Cotxe cotxe : cotxes) {
            System.out.println(cotxe.getMarca() + cotxe.getModel() + cotxe.getEstoc() + cotxe.getPreu());
        }

    }

    public String findCoxeDetallByModel(String model) {
        Assert.notNull(model);
        String result = null;
        for (Cotxe cotxe : cotxes) {
            if (model.equals(cotxe.getModel())) {
                result = "CATÀLAG DE COTXES" + "\r\n" +
                    "El Model és: " +cotxe.getModel() + "\r\n" +
                    "La marca és: "+cotxe.getMarca() + "\r\n" +
                    "L'estoc és: "+cotxe.getEstoc()+ "\r\n" +
                    "El preu és " +cotxe.getPreu()+ "\r\n";
            }
        }
        return result;
    }
}

```

```
public List<Cotxe> findCoxesByModel(String model) {

    List<Cotxe> resultats = new ArrayList<Cotxe>();

    for (Cotxe cotxe : cotxes) {
        if (cotxe.getModel().equals(model)) {
            resultats.add(cotxe);
        }
    }
    for (Cotxe cotxe : resultats) {
        System.out.println(cotxe.getMarca() + cotxe.getModel() + cotxe.getEstoc() + cotxe.getPreu());
    }

    return resultats;
}

public List<Cotxe> findCotxesByMarca(String marca) {
    List<Cotxe> resultats = new ArrayList<Cotxe>();
    for (Cotxe cotxe : cotxes) {
        if (cotxe.getMarca().equals(marca)) {
            resultats.add(cotxe);
        }
    }
    //for (Cotxe cotxe : resultats) {
        // System.out.println(cotxe.getModel() + cotxe.getMarca() + cotxe.getEstoc() + cotxe.getPreu());
    //}

    return resultats;
}

public List<Cotxe> obtenirTotsElsCotxes() {
    List<Cotxe> resultats = new ArrayList<Cotxe>();
    for (Cotxe cotxe : cotxes) {
        resultats.add(cotxe);
        //System.out.println(cotxe.getModel() + cotxe.getMarca() + cotxe.getEstoc() + cotxe.getPreu());
    }

    return resultats;
}

}
```

---

//CotxeEndpoint



```
import cat.xtec.ioc.domain.repository.impl.InMemoryCotxeRepository;
import ioc.xtec.cat.domain.cotxe.Cotxe;
import ioc.xtec.cat.domain.cotxe.services.GetCatalagCotxesByModelRequest;
import ioc.xtec.cat.domain.cotxe.services.GetCatalagCotxesByModelResponse;
import ioc.xtec.cat.domain.cotxe.services.GetCatalagCotxesDetallByModelRequest;
import ioc.xtec.cat.domain.cotxe.services.GetCatalagCotxesDetallByModelResponse;
import ioc.xtec.cat.domain.cotxe.services.GetLlistatCotxesByMarcaRequest;
import ioc.xtec.cat.domain.cotxe.services.GetLlistatCotxesByMarcaResponse;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.ws.server.endpoint.annotation.Endpoint;
import org.springframework.ws.server.endpoint.annotation.PayloadRoot;
import org.springframework.ws.server.endpoint.annotation.RequestPayload;
import org.springframework.ws.server.endpoint.annotation.ResponsePayload;
@Endpoint
public class CotxeEndpoint {

    private static final String NAMESPACE_URI = "http://cat.xtec.ioc/domain/cotxe/services";
    private InMemoryCotxeRepository cotxeRepository;
    @Autowired
    public CotxeEndpoint(InMemoryCotxeRepository cotxeRepository) {
        this.cotxeRepository = cotxeRepository;
    }

    @PayloadRoot(namespace = NAMESPACE_URI, localPart = "getCatalagCotxesDetallByModelRequest")
    @ResponsePayload
    public GetCatalagCotxesDetallByModelResponse getCotxesDetallByModel(@RequestPayload GetCatalagCotxesDetallByModelRequest request) {
        GetCatalagCotxesDetallByModelResponse response = new GetCatalagCotxesDetallByModelResponse();
        response.setCatalagCotxesDetallByModel(cotxeRepository.findCoxeDetallByModel(request.getModel()));
        return response;
    }

    @PayloadRoot(namespace = NAMESPACE_URI, localPart = "getCatalagCotxesByModelRequest")
    @ResponsePayload
    public GetCatalagCotxesByModelResponse getCotxesByModel(@RequestPayload GetCatalagCotxesByModelRequest request){

        GetCatalagCotxesByModelResponse response=new GetCatalagCotxesByModelResponse();

        for(Cotxe cotxe:cotxeRepository.findCoxesByModel(request.getModel())){
            response.getCatalagCotxesByModel().add(cotxe);
        }
        return response;
    }

    @PayloadRoot(namespace = NAMESPACE_URI, localPart = "getLlistatCotxesByMarcaRequest")
    @ResponsePayload
    public GetLlistatCotxesByMarcaResponse getCotxesByMarca(@RequestPayload GetLlistatCotxesByMarcaRequest request) {
        GetLlistatCotxesByMarcaResponse response = new GetLlistatCotxesByMarcaResponse();
        //response.setCompany(companyRepository.findCompany(request.getCif()));
        for(Cotxe cotxe:cotxeRepository.findCotxesByMarca(request.getMarca())){
            response.getLlistatCotxesByMarca().add(cotxe);
            //System.out.print("hola mundo");
        }
        return response;
    }
}
```

Comentari:

Imaginem que ja tenim el servei web anterior del que hem escrit el contracte desplegat i funcionant.

Escriviu el missatge SOAP que hem d'enviar si volem executar la consulta dels cotxes d'un mateix fabricant.

Escriviu també la resposta que ens enviarà el servei.

//En la terminar fem servir la comanda:

// Com que la resta de codi es pot veure en la pregunta 1, aquí únicament s'afegirà els xml

`curl --header "content-type: text/xml" -d @request.xml http://localhost:8080/soapws/`

//request.xml

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"  
xmlns:ser="http://cat.xtec.ioc/domain/cotxe/services">  
 <soapenv:Header/>  
 <soapenv:Body>  
 <ser:getLlistatCotxesByMarcaRequest>  
 <ser:marca>nissan</ser:marca>  
 </ser:getLlistatCotxesByMarcaRequest>  
 </soapenv:Body>  
</soapenv:Envelope>

//response.xml<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">  
 <SOAP-ENV:Header/>  
 <SOAP-ENV:Body>  
 <ns3:getLlistatCotxesByMarcaResponse xmlns:ns2="http://cat.xtec.ioc/domain/cotxe"  
xmlns:ns3="http://cat.xtec.ioc/domain/cotxe/services">  
 <ns3:llistatCotxesByMarca>  
 <ns2:model>LEAF</ns2:model>  
 <ns2:marca>nissan</ns2:marca>  
 <ns2:estoc>50</ns2:estoc>  
 <ns2:preu>20000.0</ns2:preu>  
 </ns3:llistatCotxesByMarca>  
 <ns3:llistatCotxesByMarca>  
 <ns2:model>Micra</ns2:model>  
 <ns2:marca>nissan</ns2:marca>  
 <ns2:estoc>500</ns2:estoc>  
 <ns2:preu>25000.0</ns2:preu>  
 </ns3:llistatCotxesByMarca>  
 </ns3:getLlistatCotxesByMarcaResponse>  
 </SOAP-ENV:Body>  
</SOAP-ENV:Envelope>

Comentari:

3

Puntuació 1,00 sobre 1,00

Correcte

Quines de les següents anotacions són proporcionades per Spring-WS?

Trieu-ne una o més:

- ☒ a. @PayloadRoot ✓
- ☐ b. @Webservice
- ☐ c. @Repository
- ☒ d. @Endpoint ✓

La resposta és correcta.

Les respostes correctes són: @PayloadRoot, @Endpoint

4

Puntuació 1,00 sobre 1,00

Correcte

El desenvolupament d'un WS amb Spring només es permet en la modalitat Contract-First

Trieu-ne una:

- ☒ Vertader ✓
- ☐ Fals

La resposta correcta és 'Vertader'.

Imaginem que ja tenim el servei web anterior del que hem escrit el contracte desplegat i funcionant.

Escriviu el JSON que mostrarà la petició a la URL d'una hipotètica operació que retorni tots els Cotxes. Per ser coherent, mostreu el codi del mètode initData que implementàreu al repositori i el JSON

```

//InMemoryCotxeRepository
import cat.xtec.ioc.domain.Cotxe;
import java.util.ArrayList;
import java.util.List;
import javax.annotation.PostConstruct;
import org.springframework.stereotype.Repository;
@Repository
public class InMemoryCotxeRepository {
    private static final List<Cotxe> cotxes = new ArrayList<Cotxe>();
    @PostConstruct
    public void initData() {
        Cotxe nissan = new Cotxe();
        nissan.setMarca("nissan");
        nissan.setModel("LEAF");
        nissan.setEstoc(50);
        nissan.setPreu(2000f);
        cotxes.add(nissan);

        Cotxe nissan2 = new Cotxe();
        nissan2.setMarca("nissan");
        nissan2.setModel("Micra");
        nissan2.setEstoc(500);
        nissan2.setPreu(25000f);
        cotxes.add(nissan2);

        Cotxe tesla = new Cotxe();
        tesla.setMarca("Tesla");
        tesla.setModel("Model S");
        tesla.setEstoc(200);
        tesla.setPreu(75000f);
        cotxes.add(tesla);
        Cotxe ford = new Cotxe();
        ford.setMarca("Ford");
        ford.setModel("B-Max");
        ford.setEstoc(300);
        ford.setPreu(30000f);
        cotxes.add(ford);
        for (Cotxe cotxe : cotxes) {
            System.out.println(cotxe.getMarca() + cotxe.getModel() + cotxe.getEstoc() + cotxe.getPreu());
        }
    }
}
/*
public String findCoxeByModel(String model) {
    Assert.notNull(model);
    String result = null;
    for (Cotxe cotxe : cotxes) {
        if (model.equals(cotxe.getModel())) {
            result = "El Model es: "+cotxe.getModel() +" La marca es: " + cotxe.getMarca() + " El estoc es: " +
cotxe.getEstoc()+ " El preu es " + cotxe.getPreu();
        }
    }
    return result;
}
*/
public List<Cotxe> findCoxesByModel(String model) {
    List<Cotxe> resultats = new ArrayList<Cotxe>();

    for (Cotxe cotxe : cotxes) {
        if (cotxe.getModel().equals(model)) {
            resultats.add(cotxe);
        }
    }
    for (Cotxe cotxe : resultats) {
        System.out.println(cotxe.getMarca() + cotxe.getModel() + cotxe.getEstoc() + cotxe.getPreu());
    }
    return resultats;
}

public List<Cotxe> findCotxesByMarca(String marca) {
    List<Cotxe> resultats = new ArrayList<Cotxe>();
    for (Cotxe cotxe : cotxes) {
        if (cotxe.getMarca().equals(marca)) {
            resultats.add(cotxe);
        }
    }
    //for (Cotxe cotxe : resultats) {
    //    System.out.println(cotxe.getModel() + cotxe.getMarca() + cotxe.getEstoc() + cotxe.getPreu());
    //}
}

```

```
        return resultats;
    }

    public List<Cotxe> obtenirTotsElsCotxes() {
        List<Cotxe> resultats = new ArrayList<Cotxe>();
        for (Cotxe cotxe : cotxes) {
            resultats.add(cotxe);
            //System.out.println(cotxe.getModel() + cotxe.getMarca() + cotxe.getEstoc() + cotxe.getPreu());
        }
        return resultats;
    }
}

// CotxeController
import cat.xtec.ioc.domain.Cotxe;
import cat.xtec.ioc.domain.repository.impl.InMemoryCotxeRepository;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;
@RestController
@RequestMapping("/cotxes")
public class CotxeController {
    @Autowired
    private InMemoryCotxeRepository inMemoryCotxeRepository;
    public CotxeController() {
    }
    public CotxeController(InMemoryCotxeRepository inMemoryCotxeRepository) {
        this.inMemoryCotxeRepository = inMemoryCotxeRepository;
    }
    @RequestMapping(method = RequestMethod.GET)
    public @ResponseBody
    List<Cotxe> getAllCotxes() {
        return this.inMemoryCotxeRepository.obtenirTotsElsCotxes();
    }
}

//JSON
```

```
[{"model":"LEAF","marca":"nissan","estoc":50,"preu":2000.0},
```

```
{"model":"Micra","marca":"nissan","estoc":500,"preu":25000.0},
```

```
{"model":"Model S","marca":"Tesla","estoc":200,"preu":75000.0},
```

```
{"model":"B-Max","marca":"Ford","estoc":300,"preu":30000.0}]
```

Comentari:

Del nostre WS de la web de cotxes, imagineu que tenim ara implementada l'opció d'afegir-ne un de nou, mitjançant un POST.

Escriviu el JSON que utilitzaríeu per afegir aquesta informació invocant al WS utilitzant, per exemple, curl

```
//curl:
curl -H "Content-Type: application/json" -X POST -d "
{"model\":\"model_afegit\",\"marca\":\"marca_afegida\",\"estoc\":\"24\",\"preu\":\"10000\"}"
http://localhost:8080/EAC6CatalagCotxesRest/cotxes
//JSON:
[
  {
    "model": "LEAF",
    "marca": "nissan",
    "estoc": 50,
    "preu": 2000
  },
  {
    "model": "Micra",
    "marca": "nissan",
    "estoc": 500,
    "preu": 25000
  },
  {
    "model": "Model S",
    "marca": "Tesla",
    "estoc": 200,
    "preu": 75000
  },
  {
    "model": "B-Max",
    "marca": "Ford",
    "estoc": 300,
    "preu": 30000
  },
  {
    "model": "model_afegit",
    "marca": "marca_afegida",
    "estoc": 24,
    "preu": 10000
  }
]
```

```

//Controller: CotxeController
import cat.xtec.ioc.domain.Cotxe;
import cat.xtec.ioc.domain.repository.impl.InMemoryCotxeRepository;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;
@RestController
@RequestMapping("/cotxes")
public class CotxeController {
    @Autowired
    private InMemoryCotxeRepository inMemoryCotxeRepository;
    public CotxeController() {
    }
    public CotxeController(InMemoryCotxeRepository inMemoryCotxeRepository) {
        this.inMemoryCotxeRepository = inMemoryCotxeRepository;
    }

    @RequestMapping(value = "{marca}", method = RequestMethod.GET)
    public @ResponseBody
    Cotxe getByMarca(@PathVariable String marca) {
        return this.inMemoryCotxeRepository.obtenirCotxePerMarca(marca);
    }
    @RequestMapping(method = RequestMethod.GET)
    public @ResponseBody
    List<Cotxe> getAllCotxes() {
        return this.inMemoryCotxeRepository.obtenirTotsElsCotxes();
    }

    @RequestMapping(method = RequestMethod.POST)
    public @ResponseBody
    ResponseEntity<Cotxe> create(@RequestBody Cotxe cotxe) {
        this.inMemoryCotxeRepository.afigir(cotxe);
        return new ResponseEntity<>(cotxe, HttpStatus.CREATED);
    }
}

```

Comentari:

Tècnicament el JSON porta massa informació. Al POST només cal posar-hi la informació nova



**JAX-RS és una API...**

Trieu-ne una:

- ☐ a. serveix per desenvolupar serveis web
- ☐ b. cap resposta és correcta
- ☒ c. només per desenvolupar serveis RESTFul ✓
- ☐ d. pot servir per desenvolupar serveis RESTFul, però també WS

La resposta és correcta.

La resposta correcta és: només per desenvolupar serveis RESTFul

**És certa la següent afirmació?**

Spring proporciona un mòdul específic per a la creació i el consum de serveis web RESTful.

Trieu-ne una:

- ☐ Vertader
- ☒ Fals ✓

La resposta correcta és 'Fals'.

Imagineu que tenim implementat el WS de cotxes del que hem parlat amb anterioritat.

Escriviu una classe de test per tal de provar el ws, que està publicat a la URL `http://localhost:8080/cotxes/all` i que retorna TOTS els cotxes que tenim al sistema, que assegurí que la resposta és un JSON.

Escriviu també la classe de dades que utilitzeu. Recordeu utilitzar correctament les anotacions pertinents.

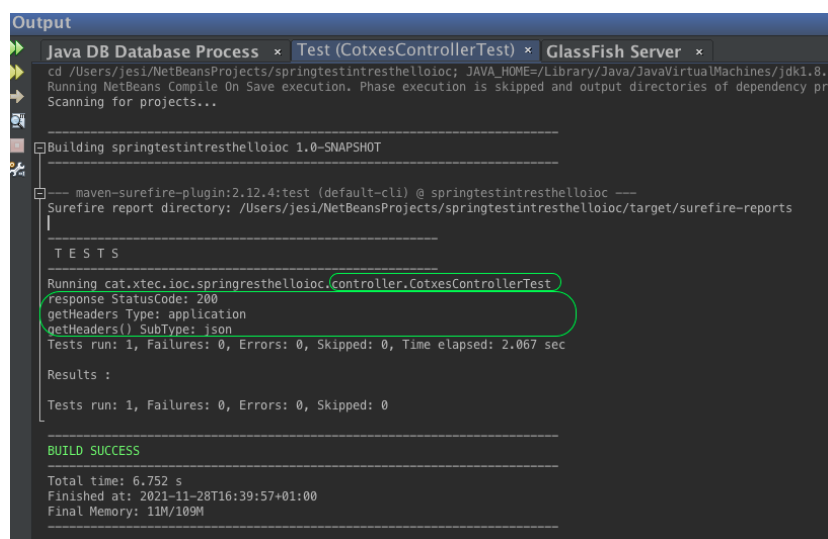
```
//CotxesControllerTest
import javax.ws.rs.core.MediaType;
import static org.junit.Assert.assertEquals;
import org.junit.Test;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.client.RestTemplate;

public class CotxesControllerTest {
    private static final String uri = "http://localhost:8080/EAC6CatalagCotxesRest/cotxes/all";
    @Test
    public void getCotxesShouldReturnOKAndJSON () {
        // Arrange
        RestTemplate restTemplate = new RestTemplate();
        // Act
        ResponseEntity<CotxesTest[]> response = restTemplate.getForEntity(uri, CotxesTest[].class);
        // Assert
        assertEquals(HttpStatus.OK, response.getStatusCode());
        assertEquals(MediaType.APPLICATION_JSON,
            response.getHeaders().getContentType().getType() + "/" +
            response.getHeaders().getContentType().getSubtype());
    }
}
```

```
//response StatusCode: 200
System.out.println("response StatusCode: " + response.getStatusCode());
// getHeaders Type: application
System.out.println("getHeaders Type: "+ response.getHeaders().getContentType().getType());
//getHeaders SubType: json
System.out.println("getHeaders() SubType: "+ response.getHeaders().getContentType().getSubtype());
}
}
//CotxesTest
import com.fasterxml.jackson.annotation.JsonIgnoreProperties;

@JsonIgnoreProperties(ignoreUnknown = true)
public class CotxesTest {
    private String model;
    private String marca;
    private Integer estoc;
    public String getModel() {
        return model;
    }
    public void setModel(String model) {
        this.model = model;
    }
    public String getMarca() {
        return marca;
    }
    public void setMarca(String marca) {
        this.marca = marca;
    }
    public Integer getEstoc() {
        return estoc;
    }
    public void setEstoc(Integer estoc) {
        this.estoc = estoc;
    }
    public Float getPreu() {
        return preu;
    }
    public void setPreu(Float preu) {
        this.preu = preu;
    }
    private Float preu;
    @Override
    public String toString() {
        return "CotxesTest{" + "model=" + model + ", marca=" + marca + ", estoc=" + estoc + ", preu=" + preu + '}';
    }
}
```

```
//getAll()
@RequestMapping(method = RequestMethod.GET, value = "/all")
List<Cotxe> getAll() {
    return this.inMemoryCotxeRepository.obtenirTotsElsCotxes();
}
```



The screenshot shows an IDE output window with the following content:

```
Output
Java DB Database Process x Test (CotxesControllerTest) x GlassFish Server x
cd /Users/jesi/NetBeansProjects/springtestintresthelloioc; JAVA_HOME=/Library/Java/JavaVirtualMachines/jdk1.8.0_101.jdk/Contents/Home;
Running NetBeans Compile On Save execution. Phase execution is skipped and output directories of dependency projects are scanned.
Scanning for projects...

[Building springtestintresthelloioc 1.0-SNAPSHOT]

[Running cat.xtec.ioc.springresthelloioc.Controller.CotxesControllerTest]
response StatusCode: 200
getHeaders Type: application
getHeaders() SubType: json
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 2.067 sec

Results :
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

BUILD SUCCESS

Total time: 6.752 s
Finished at: 2021-11-28T16:39:57+01:00
Final Memory: 11M/109M
```

Comentari:

L'encarregat que usa Spring per transformar el model de dades a un JSON i a l'inrevés s'anomena:

Triu-ne una:

- ☒ a. Jackson ✓
- ☐ b. Jersey
- ☐ c. JAX-WS
- ☐ d. JAX-RS

La resposta és correcta.

La resposta correcta és: Jackson