

```
1 package org.example;
2
3 import org.apache.catalina.Context;
4 import org.apache.catalina.LifecycleException;
5 import org.apache.catalina.Wrapper;
6 import org.apache.catalina.startup.Tomcat;
7 import org.example.web.MyInvoiceServlet;
8
9 public class ApplicationLauncher {
10
11     static final int defPort = 8080;
12
13
14     public static void main(String[] args) throws
LifecycleException {
15         /** use this with java -Dport80xx -jar
target/xy
16         if Port not provided as Parameter -> use
default Port */
17
18         int port;
19         String sPort = (System.getProperty("port"
));
20
21         if ((sPort == null) || (sPort.isEmpty())) {
22             port = defPort;
23         } else {
24             port = Integer.parseInt(System.
getProperty("port"));
25         }
26         Tomcat tomcat = new Tomcat();
27
28         tomcat.setPort(port);
29         tomcat.getConnector();
30
31         Context ctx = tomcat.addContext("", null);
32         Wrapper servlet = Tomcat.addServlet(ctx, "
org.example.web.MyInvoiceServlet", new
MyInvoiceServlet());
33         servlet.setLoadOnStartup(1);
34         servlet.addMapping("/*");
35
36         tomcat.start();
37         System.out.println("Tomcat started.....");
```

```
38         System.out.println("Listen at Port: " +  
    port);  
39     }  
40 }
```

```
1 package org.example.web;
2
3 import com.fasterxml.jackson.databind.ObjectMapper;
4 import org.example.context.ApplicationConfiguration
5 ;
6 import org.example.model.Invoice;
7 import org.example.service.InvoiceService;
8 import org.example.service.UserService;
9 import org.springframework.context.annotation.
10 AnnotationConfigApplicationContext;
11
12 import javax.servlet.ServletException;
13 import javax.servlet.http.HttpServlet;
14 import javax.servlet.http.HttpServletRequest;
15 import javax.servlet.http.HttpServletResponse;
16 import java.io.IOException;
17 import java.util.List;
18
19 public class MyInvoiceServlet extends HttpServlet {
20     private UserService userService;
21     private ObjectMapper objectMapper;
22     private InvoiceService invoiceService;
23
24     @Override
25     public void init() throws ServletException{
26         AnnotationConfigApplicationContext ctx
27             = new
28             AnnotationConfigApplicationContext(
29             ApplicationConfiguration.class);
30
31         // is only needed that @PreDestroy works
32         properly with IDE
33         ctx.registerShutdownHook();
34
35         this.userService = ctx.getBean(UserService.
36         class);
37         this.objectMapper = ctx.getBean(
38         ObjectMapper.class);
39         this.invoiceService = ctx.getBean(
40         InvoiceService.class);
41
42         System.out.println("im servlet - init");
43     }
44 }
```

```

37     @Override
38     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
        IOException{
39         if (request.getRequestURI().
            equalsIgnoreCase("/invoices")) {
40             String userId = request.getParameter("
                user_id");
41             Integer amount = Integer.valueOf(
                request.getParameter("amount"));
42             String helloMsg = "Hello, My Dear";
43
44             Invoice invoice = invoiceService.create
                (userId, amount, helloMsg);
45
46             response.setContentType("application/
                json; charset=UTF-8");
47             String json = new ObjectMapper().
                writeValueAsString(invoice);
48             response.getWriter().print(json);
49         }
50         else {
51             response.setStatus(HttpServletResponse.
                SC_NOT_FOUND);
52         }
53     }
54     @Override
55     protected void doGet(HttpServletRequest request
        ,HttpServletResponse response) throws IOException {
56         if (request.getRequestURI().
            equalsIgnoreCase("/")) {
57             response.setContentType("text/html;
                charset=UTF-8");
58             response.getWriter().print(
59                 "<html>\n" +
60                 "<body>\n" +
61                 "<h1>Hello Barbara</h1>
                \n" +
62                 "<p>You are working on
                your first Spring App</p>\n" +
63                 "</body>\n" +
64                 "</html>\n");
65         }
66         else if (request.getRequestURI().

```

```
66 equalsIgnoreCase("/invoices"))
67     {
68         response.setContentType("application/
json; charset=UTF-8");
69         List<Invoice> invoices =
invoiceService.findAll();
70
71         System.out.println(objectMapper.
writeValueAsString(invoices));
72         // (2)
73         response.getWriter().print(
objectMapper.writeValueAsString(invoices));
74         // (3)
75     }
76 }
77 }
78
```

```
1 package org.example.model;
2
3 public class User {
4
5     private String id;
6     private String name;
7
8     public User(){
9     }
10
11     public User(String id, String name) {
12         this.id = id;
13         this.name = name;
14     }
15     public String getId() {
16         return id;
17     }
18     public void setId(String id) {
19         this.id = id;
20     }
21
22     public String getName() {
23         return name;
24     }
25     public void setName(String name) {
26         this.name = name;
27     }
28 }
29
```

```
1 package org.example.model;
2
3 import com.fasterxml.jackson.annotation.
  JsonProperty;
4 import java.util.UUID;
5
6 public class Invoice {
7     private String id;
8
9     @JsonProperty("user_id")
10    private String userId;
11
12    @JsonProperty("pdf_url")
13    private String pdfUrl;
14
15    private Integer amount;
16    // ist nur als Test hinzugefügt worden
17    private String myHello;
18
19    public Invoice(){
20    }
21
22    public Invoice(String userId, Integer amount,
23    String pdfUrl,String myHello) {
24        this.id = UUID.randomUUID().toString();
25        this.userId = userId;
26        this.pdfUrl = pdfUrl;
27        this.amount = amount;
28        this.myHello = myHello;
29    }
30
31    public String getId() {
32        return id;
33    }
34
35    public void setId(String id) {
36        this.id = id;
37    }
38
39    public String getUserId() {
40        return userId;
41    }
42
43    public void setUserId(String userId) {
44        this.userId = userId;
45    }
46
47    }
```

```
43
44     public String getPdfUrl(){
45         return pdfUrl;
46     }
47     public void setPdfUrl(String pdfUrl){
48         this.pdfUrl = pdfUrl;
49     }
50
51     public Integer getAmount() {
52         return amount;
53     }
54     public void setAmount(Integer amount){
55         this.amount = amount;
56     }
57
58     public String getMyHello() { return myHello; }
59     public void setMyHello(String myHello){
60         this.myHello = myHello;
61     }
62 }
63
```



```
1 package org.example.context;
2
3 import com.fasterxml.jackson.databind.ObjectMapper;
4 import org.example.ApplicationLauncher;
5 import org.example.service.InvoiceService;
6 import org.example.service.UserService;
7 import org.springframework.beans.factory.config.
    ConfigurableBeanFactory;
8 import org.springframework.context.
    ConfigurableApplicationContext;
9 import org.springframework.context.annotation.*;
10
11 @Configuration
12 @ComponentScan(basePackageClasses =
    ApplicationLauncher.class)
13 // Reihenfolge ist wichtig (overwrite) & falls File
    nicht vorhanden ignore
14 @PropertySource("classpath:/application.properties"
    )
15 @PropertySource(value="classpath:/application-${
    spring.profiles.active}.properties"
16     , ignoreResourceNotFound = true)
17
18 public class ApplicationConfiguration {
19
20     @Bean
21     public ObjectMapper objectMapper(){
22         return new ObjectMapper();
23     }
24 }
25
```

```
1 package org.example.service;
2
3 import org.example.model.User;
4 import org.springframework.stereotype.Component;
5
6 import java.util.UUID;
7
8 @Component
9 public class UserService {
10
11     public User findById(String id) {
12         String randomName = UUID.randomUUID().
13         toString();
14         // always "finds" the user, every user has
15         a random name
16         return new User(id,randomName);
17     }
18 }
```

```
1 package org.example.service;
2
3 import org.example.model.Invoice;
4 import org.example.model.User;
5 import org.springframework.beans.factory.annotation
    .Value;
6 import org.springframework.stereotype.Component;
7
8 import javax.annotation.PostConstruct;
9 import javax.annotation.PreDestroy;
10 import java.util.List;
11 import java.util.concurrent.CopyOnWriteArrayList;
12
13 @Component
14 public class InvoiceService {
15
16     private final UserService userService;
17     private final List<Invoice> invoices = new
    CopyOnWriteArrayList<>(); // (1)
18     private final String cdnUrl;
19     private final String myMsg;
20
21     public InvoiceService(UserService userService
    , @Value("${cdn.url}") String cdnUrl, @Value("${my.
    hello}") String myMsg) {
22         this.userService = userService;
23         this.cdnUrl = cdnUrl;
24         this.myMsg = myMsg;
25         System.out.println(this.myMsg);
26     }
27
28     // in old Spring Versions or if more than one
    constructor: @Autowired required
29     /**
30     public InvoiceService(UserService userService
    ) {
31         System.out.println("Initialisierung
    Konstruktor InvoiceService");
32         this.userService = userService;
33     }
34     */
35
36     @PostConstruct
37     public void init() {
```

```

38         System.out.println("Fetch PDF Template from
        S3...");
39         // TODO download from s3 and save locally
40     }
41     @PreDestroy
42     // works only when application is really
    shutdown
43     public void shutdown(){
44         System.out.println("Deleting downloaded
    template");
45     }
46     // ----- Start of the methods -----
47
48     public List<Invoice> findAll() {
49         // Debugging : System.out.println("in
    findAll");
50         return invoices;
51     }
52     public Invoice create(String userId, Integer
    amount, String myMsg){
53         User user = userService.findById(userId);
54         if (user == null) {
55             throw new IllegalStateException();
56         }
57         // TODO real pdf creation and storing it on
    network server
58         // Invoice invoice = new Invoice(userId,
    amount, "http://www.africau.edu/images/default/
    sample.pdf");
59
60         Invoice invoice = new Invoice(userId, amount
    , cdnUrl+"/images/default/sample.pdf", myMsg);
61         /* Debugging old style
62         System.out.print("Create org.example.model.
    Invoice: ");
63         System.out.println(invoice);
64         */
65         invoices.add(invoice);
66         return invoice;
67     }
68 }

```

```
1 package org.example.service;
2
3 import org.springframework.context.annotation.
  Profile;
4 import org.springframework.stereotype.Service;
5
6 import javax.annotation.PostConstruct;
7
8 @Service
9 @Profile("dev")
10
11 // This needs to run with following command: java -
Dspring.profiles.active=dev -jar target/MyI*
12 public class DummyInvoiceServiceLoader {
13     private final InvoiceService invoiceService;
14
15     public DummyInvoiceServiceLoader(InvoiceService
  invoiceService) {
16         this.invoiceService = invoiceService;
17     }
18
19     @PostConstruct
20     public void setup() {
21         System.out.println("Creating dev Invoices
  ...");
22         invoiceService.create("Sibylle", 50, "Hello
  , my Dear");
23         invoiceService.create("Barbara", 20, "Hello
  , my Dear");
24     }
25 }
26
```

```
1 Manifest-Version: 1.0
2 Main-Class: org.example.ApplicationLauncher
3
4
```

```
1 cdn.url=https://dev-cdn.marcobehler.com
2 my.hello=Hallo Barbara
3
```

```
1 cdn.url=https://some-dev-url.marcobehler.com  
2 my.Hello=Das ist die Vorkasse
```