Jakarta EE Cheat Sheet

CDI · JAX-RS · Servlet · JPA · EJB

SETUP & APP SERVER

```
WildFly / GlassFish / Payara /
OpenLiberty
<packaging>war</packaging>
<dep>jakarta.jakartaee-api 10</dep>
(provided)
@ApplicationPath("/api") class AppConfig
extends Application{}
```

Deploys as .war on servlet container or app server

CDI BEANS

```
@RequestScoped / @SessionScoped /
@ApplicationScoped
@Dependent (default  lifecycle follows
owner)
@Inject   field, constructor, or setter
@Named("myBean")   EL accessible
@Produces   factory method for CDI
@Qualifier   disambiguate multiple impls
```

JAX-RS ENDPOINTS

```
@Path("/users") @ApplicationScoped
public class UserResource {
@GET @Path("/{id}")
@Produces(APPLICATION_JSON)
public User getUser(@PathParam("id")
Long id){...}
@POST @Consumes(APPLICATION_JSON)
public Response create(User u){...}
}
```

SERVLETS

```
@WebServlet("/hello")
public class HelloServlet extends
HttpServlet{
protected void doGet(HttpServletRequest
req,
HttpServletResponse res){
res.getWriter().write("Hello");
}
}
```

JPA INTEGRATION

```
@PersistenceContext EntityManager em;
@Stateless  // or @RequestScoped +
@Transactional
public List<User> findAll(){
return em.createQuery("FROM
User",User.class).getResultList();
}
```

JTA transactions managed by container

BEAN VALIDATION

```
@NotNull @Size(min=2,max=50) String
name;
@Email String email;
@Min(0) int age;
@Valid in JAX-RS params triggers
validation
Validator v =
Validation.buildDefaultValidatorFactory().getValidator();
Set<ConstraintViolation<T>> errs =
v.validate(obj);
```

JMS MESSAGING

```
@Inject @JMSConnectionFactory("jms/cf")
JMSContext ctx;
@Resource(lookup="jms/queue") Queue q;
ctx.createProducer().send(q, "message");
@MessageDriven(activationConfig={...})
void onMessage(Message m){...}  // MDB
```

EJB BASICS

```
@Stateless   no state, pooled
@Stateful    conversational state per
client
@Singleton  @Startup   one instance
```

Container manages transactions on @Stateless by default

```
@TransactionAttribute(REQUIRED/REQUIRES_NEW/NOT_SUPPORTED)
```

EJBs deprecated in favor of CDI for new projects

INTERCEPTORS

```
@Interceptor @LoggingBinding
public class LoggingInterceptor{
@AroundInvoke
public Object log(InvocationContext ctx)
throws Exception{
log.info("Calling
"+ctx.getMethod().getName());
return ctx.proceed();
}
}
```

WEBSOCKET

```
@ServerEndpoint("/ws/chat")
public class ChatEndpoint{
@OnOpen void onOpen(Session s){}
@OnMessage void onMsg(String msg,
Session s){}
@OnClose void onClose(Session s){}
}
```

Full-duplex communication over HTTP upgrade

SECURITY

```
@DeclareRoles({"admin","user"})
@RolesAllowed("admin") on method or
class
@PermitAll / @DenyAll
SecurityContext.isUserInRole("admin")
@WebSecurityDef (Soteria 2.0  Jakarta
Security)
@LoginConfig(authMethod="FORM") in
web.xml
```

COMMON GOTCHAS

EJB must not be new'd directly use @Inject

CDI @RequestScoped doesn't work in async/scheduler

JAX-RS providers need @Provider on the class

persistence.xml required even with Spring-style JPA

Application server version must match API version