

JPA 2.2 Within Java EE 8



Antonio Goncalves

JAVA CHAMPION

@agoncal www.antoniogoncalves.org



Previous Module



Life cycle

Not anemic objects

Business logic

Callback annotations

Entity listeners

Overview



Java Enterprise Edition 8

Services given by the Java EE container

Java Persistence API integration

- Java EE and CDI
- Bean Validation
- Transactional components
- External services (JAX-RS or JMS)

Summary

References



Quick Overview of Java EE 8



Java Enterprise Edition

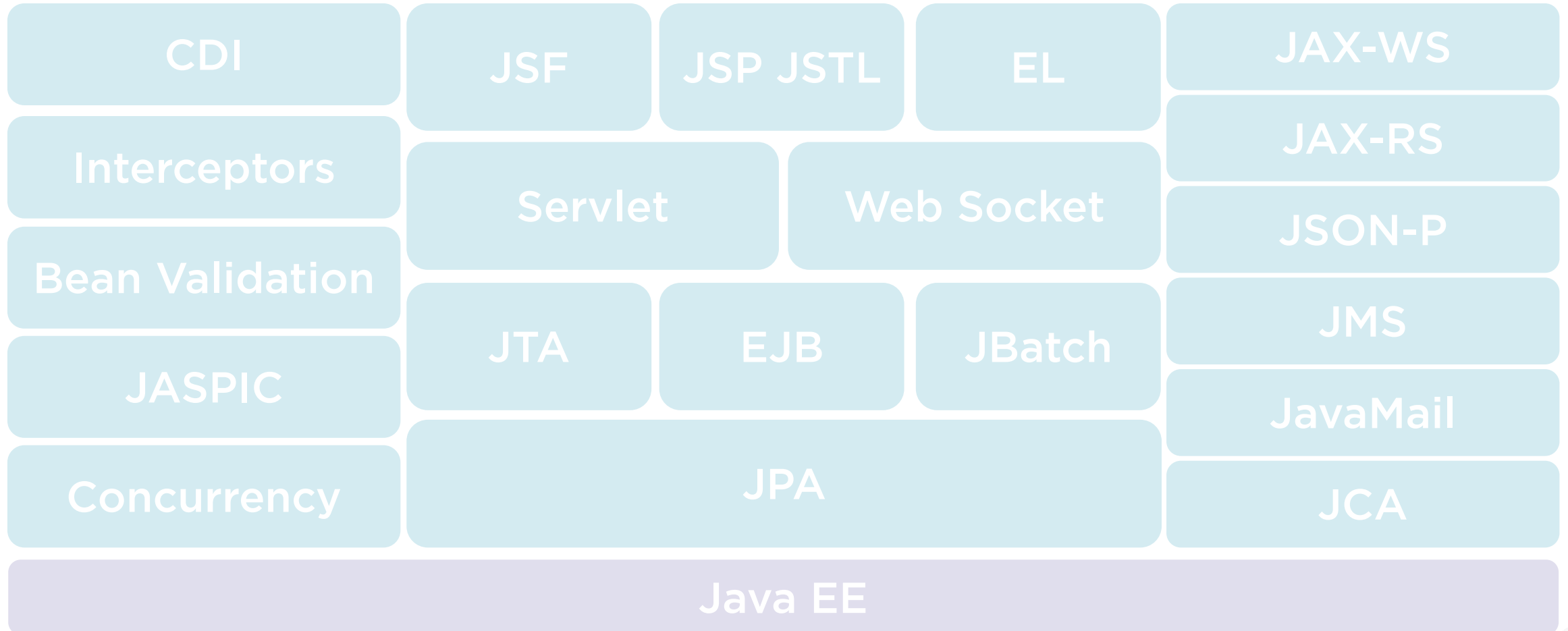
Umbrella specification

- JSR 366
- Contains 34 specifications
- Make them work together

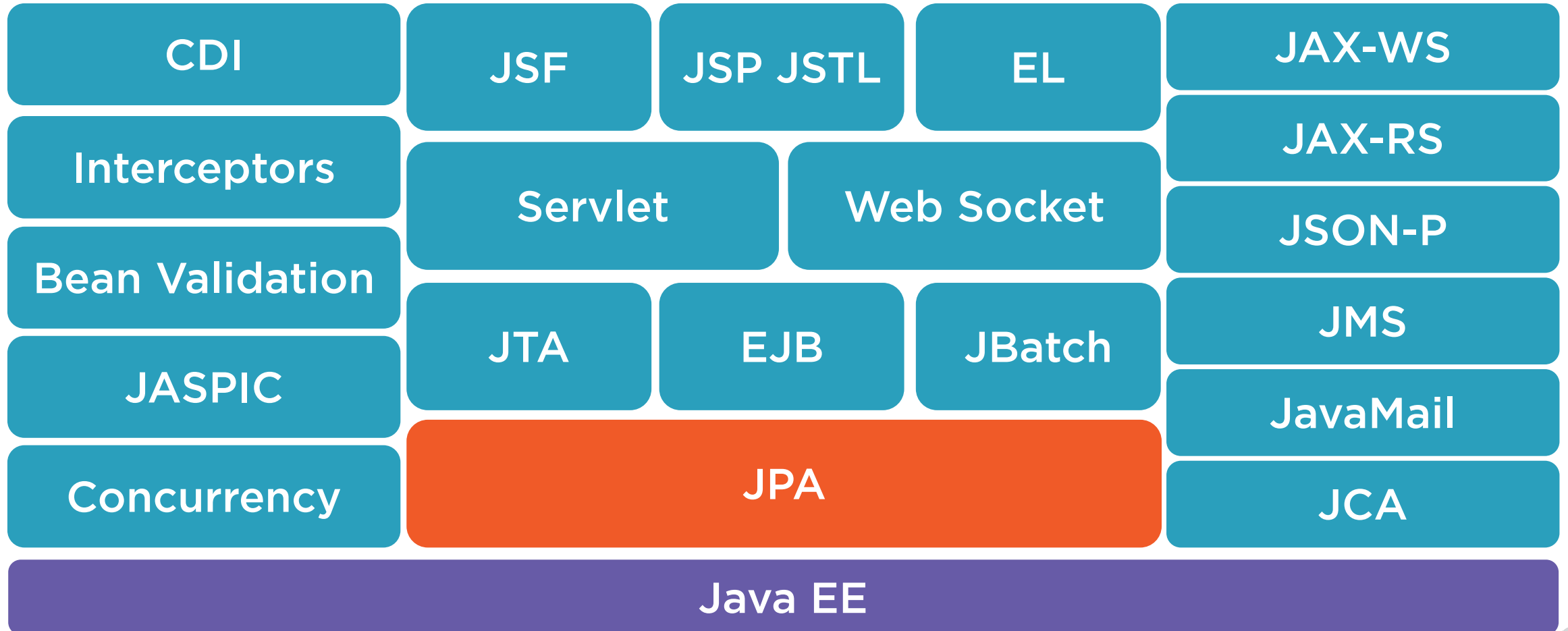
Java EE is a managed environment

- Container
- Providers

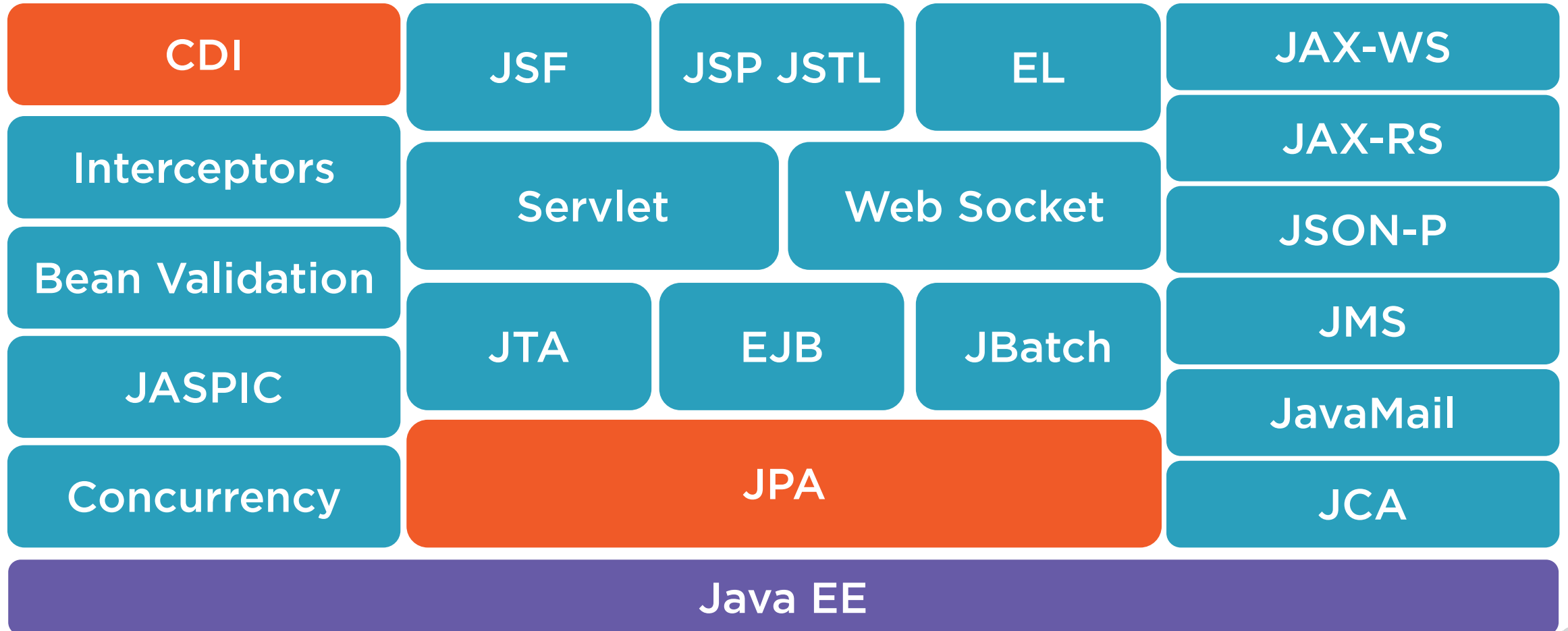
Main Java EE 8 Specifications



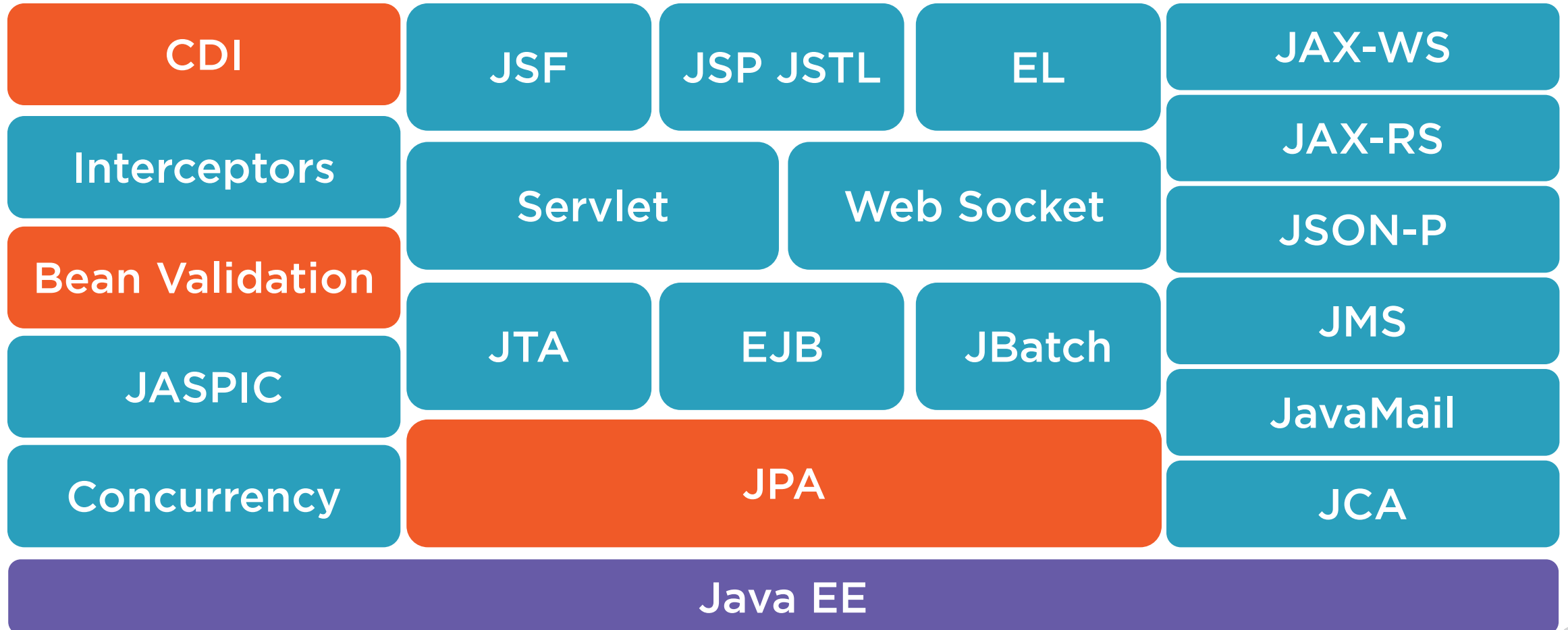
JPA Interactions in Java EE



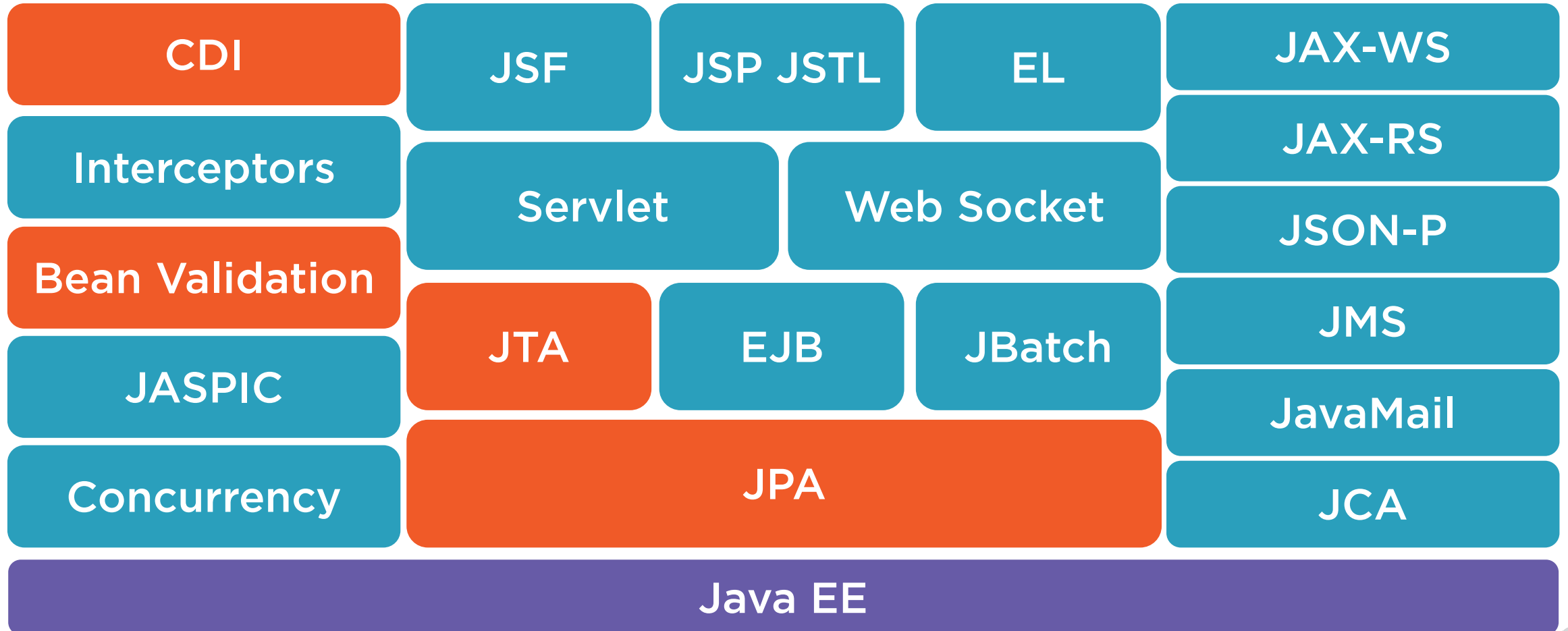
JPA Interactions in Java EE



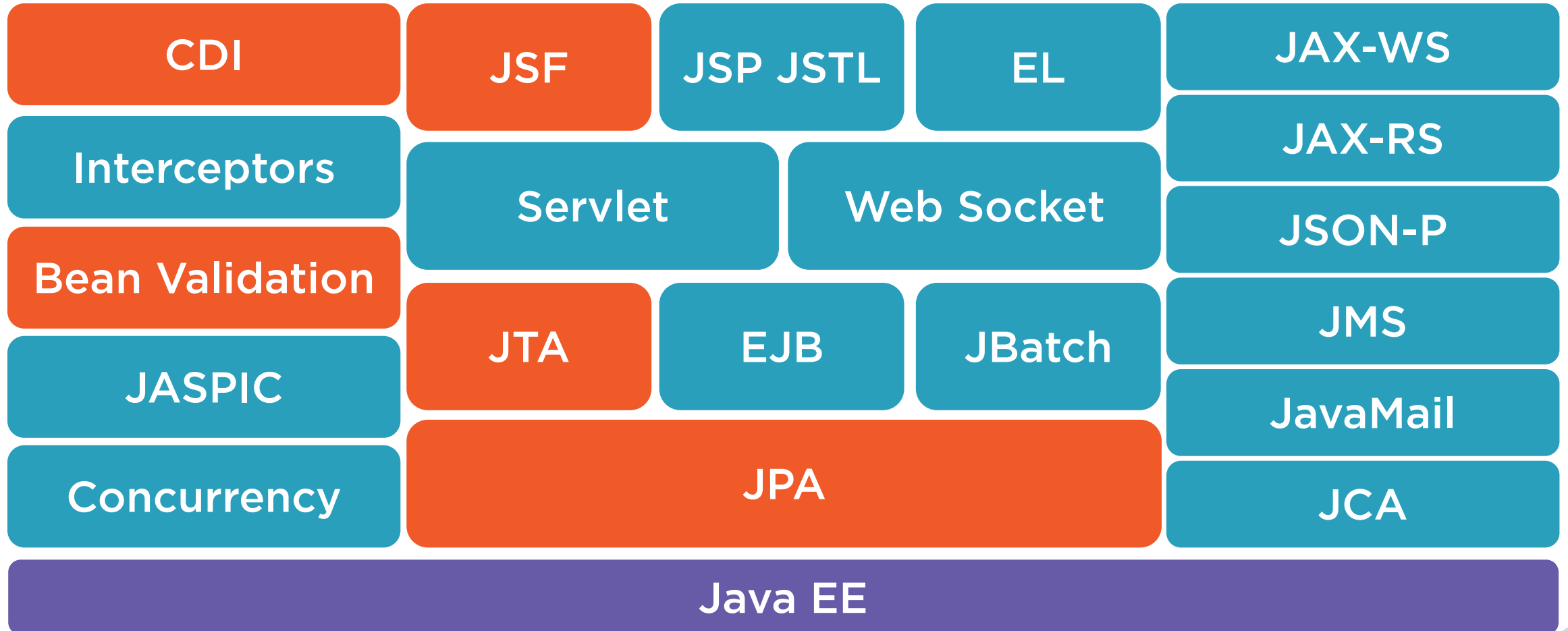
JPA Interactions in Java EE



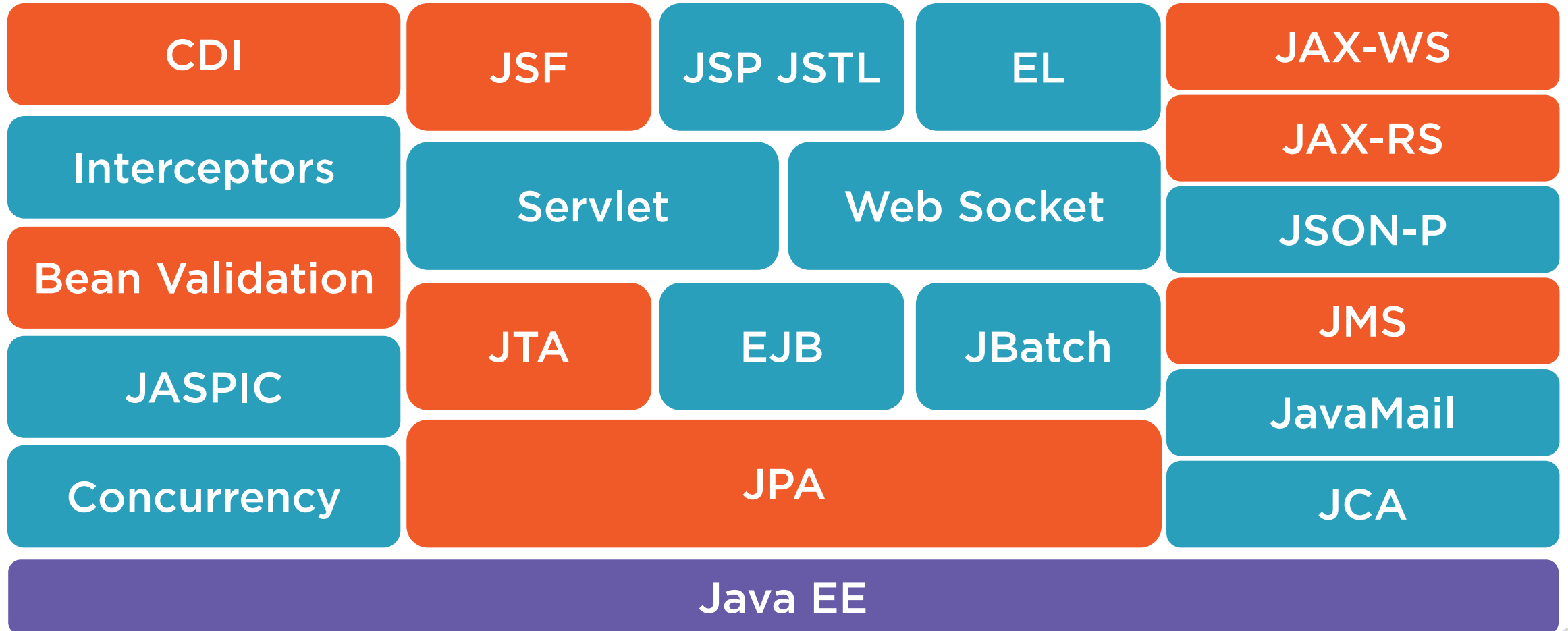
JPA Interactions in Java EE



JPA Interactions in Java EE



JPA Interactions in Java EE



Demo



CD Bookstore web application

Java EE specifications

Browse data

CRUD operations

Book entity with one-to-many Authors



Context & Dependency Injection



Integration with Java EE and CDI



Java EE container controls life-cycle

Bootstraps EntityManager

Provides access to EntityManager

- JNDI lookup
- @PersistenceContext
- @Inject

Invokes EntityManager.close()

JPA in Java SE

```
public class BookService {  
  
    public Book createBook(Book book) {  
        EntityManagerFactory emf = Persistence.createEntityManagerFactory("myPU");  
        EntityManager em = emf.createEntityManager();  
        EntityTransaction tx = em.getTransaction();  
  
        tx.begin();  
        em.persist(book);  
        tx.commit();  
        em.close();  
        emf.close();  
        return book;  
    }  
}
```



Injection of EntityManager in Java EE

```
public class BookService {  
  
    @PersistenceContext(unitName = "myPU")  
    private EntityManager em;  
  
    public Book createBook(Book book) {  
  
        EntityTransaction tx = em.getTransaction();  
  
        tx.begin();  
        em.persist(book);  
        tx.commit();  
  
        return book;  
    }  
}
```



Injection of EntityManager in Java EE with CDI

```
public class BookService {  
  
    @Inject  
    private EntityManager em;  
  
    public Book createBook(Book book) {  
  
        EntityTransaction tx = em.getTransaction();  
  
        tx.begin();  
        em.persist(book);  
        tx.commit();  
  
        return book;  
    }  
}
```



Injection of EntityManager in Java EE with CDI

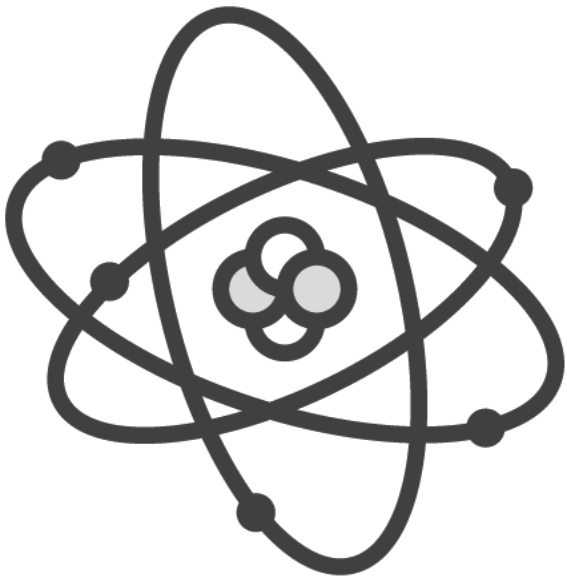
```
public class BookService {  
  
    @Inject  
    private EntityManager em;  
  
    public Book createBook(Book book) {  
  
        EntityTransaction tx = em.getTransaction();  
  
        tx.begin();  
        em.persist(book);  
        tx.commit();  
  
        return book;  
    }  
}
```



Transactions



Integration with Transactional Components



Data kept in consistent state

Groups several operations

Single unit of work

Succeed or failed atomically

JPA vs. JTA Transactions

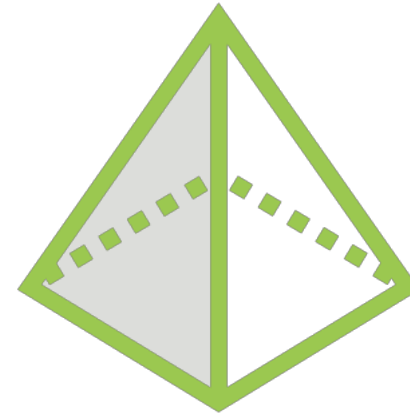


JPA EntityTransaction

Resource local transaction

Explicitly invoke commit or rollback

Not propagated



JTA UserTransaction

Resource neutral

Propagated

EntityManager cooperates with JTA



JPA EntityTransaction

```
public class BookService {  
  
    @Inject  
    private EntityManager em;  
  
    public Book createBook(Book book) {  
        EntityTransaction tx = em.getTransaction();  
        tx.begin();  
        em.persist(book);  
        tx.commit();  
        return book;  
    }  
}
```



JTA Transaction

@Transactional

```
public class BookService {
```

```
    @Inject
```

```
    private EntityManager em;
```

```
    public Book createBook(Book book) {
```

```
        em.persist(book);
```

```
        return book;
```

```
    }
```

```
}
```

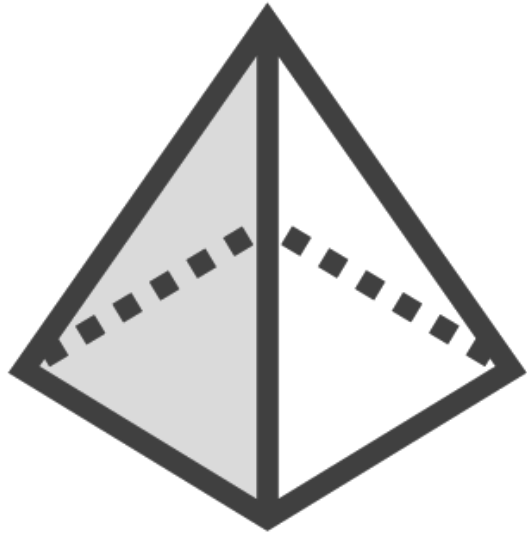
Starts a transaction



Commit or rollback the transaction



@Transactional



CDI interceptor binding

Belongs to the JTA specification

- Java Transaction API
- JSR 907

Class or method level

Controls transaction boundaries

On any Java EE managed bean

Transactional CDI Bean

@Transactional

```
public class BookService {  
  
    @Inject  
    private EntityManager em;  
  
    public Book createBook(Book book) {  
        em.persist(book);  
        return book;  
    }  
}
```



Transactional REST Endpoint

```
@Transactional
@Path("book")
public class BookService {

    @Inject
    private EntityManager em;

    @POST
    @Consumes(MediaType.APPLICATION_XML)
    public Book createBook(Book book) {
        em.persist(book);
        return book;
    }
}
```



Transactional JSF Backing Bean

```
@Transactional
@Named
public class BookService {

    @Inject
    private EntityManager em;

    public Book createBook(Book book) {
        em.persist(book);
        return book;
    }
}
```



Transactional Servlet

```
@Transactional
@WebServlet(urlPatterns = {"/TxServlet"})
public class BookService extends HttpServlet {

    @Inject
    private EntityManager em;

    public Book createBook(Book book) {
        em.persist(book);
        return book;
    }
}
```



Transactional EJB

@Stateless

```
public class BookService {
```

@Inject

```
private EntityManager em;
```

```
public Book createBook(Book book) {
```

```
    em.persist(book);
```

```
    return book;
```

```
}
```

```
}
```



Demo



Create and update a book

`@Transactional` annotation

Implicit transaction demarcation

Instead of programmatically using APIs

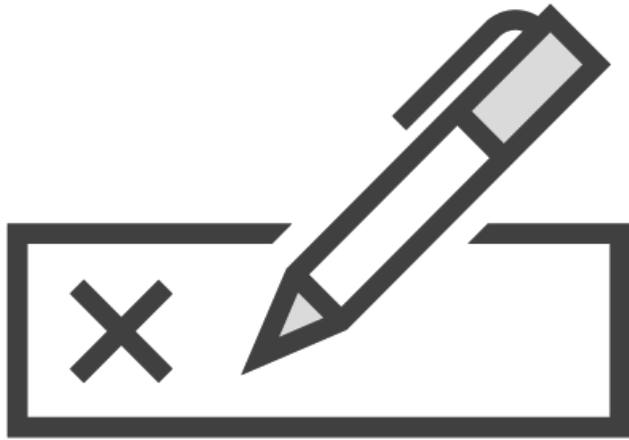
JSF Backing Bean and REST Endpoint



Bean Validation



Bean Validation



Process, store, retrieve data

Constrain our model

Ensure data is valid

The service will behave correctly

Give feedback to the users

Using annotations

Bean Validation Constraints

```
public class Artist {  
  
    private Long id;  
  
    @NotNull @Size(min = 2, max = 50)  
    private String firstName;  
  
    @NotNull @Size(min = 4, max = 60)  
    private String lastName;  
  
    @Past  
    private LocalDate dateOfBirth;
```

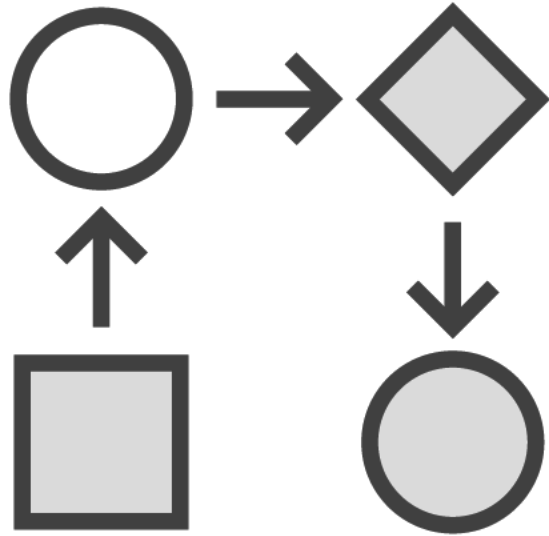


Validating Constraints

```
public class ArtistService {  
  
    @Inject  
    private Validator validator;  
  
    public void createArtist(Artist artist) {  
  
        Set<ConstraintViolation<Artist>> violations;  
        violations = validator.validate(artist);  
        if (violations.size() > 0)  
            throw new ConstraintViolationException(violations);  
    }  
}
```



Integration with Bean Validation



Entities with Bean Validation annotations

Validation automatically made

- `@PrePersist`
- `@PreUpdate`
- `@PreRemove`

Disable it in `persistence.xml`

Entity with Bean Validation Constraints

@Entity

```
public class Artist {
```

```
    @Id @GeneratedValue
```

```
    private Long id;
```

```
    @Column(name = "first_name", length = 50)
```

```
    @NotNull @Size(min = 2, max = 50)
```

```
    private String firstName;
```

```
    @Column(name = "last_name", length = 50)
```

```
    @NotNull @Size(min = 4, max = 60)
```

```
    private String lastName;
```

```
    @Past
```

```
    private LocalDate dateOfBirth;
```



Validating an Entity

```
@Transactional
public class ArtistService {

    @Inject
    private Validator validator;

    @PersistenceContext
    private EntityManager em;

    public void createArtist(Artist artist) {
        Set<ConstraintViolation<Artist>> violations;
        violations = validator.validate(artist);
        if (violations.size() > 0)
            throw new ConstraintViolationException(violations);
        em.persist(artist);
    }
}
```



Demo



Update a book with invalid data

Bean Validation constraint annotations

Book entity

Validation happens automatically

No if/then/else statement

Default error message



JAXB



Integration with JAXB



Java Architecture for XML Binding

Binds Java to XML and vice versa

Uses annotations

- @XMLRootElement, @XmlElement...

External services

- SOAP Web Service
- REST Web Service
- JMS Messages

Expose entities as XML

Entity with JAXB Annotations

```
@Entity
@XmlRootElement
public class Author {

    @Id @GeneratedValue
    @XmlAttribute
    private Long id;

    private String firstName;

    @Column(name = "last_name", length = 50)
    @XmlElement(name = "last-name")
    private String lastName;

    @Transient
    @XmlTransient
    private Integer age;
```



Exposing an Entity as XML in REST Endpoint

```
@Transactional
@Path("book")
public class BookEndpoint {

    @Inject
    private EntityManager em;

    @GET
    @Path("/{id}")
    @Produces(MediaType.APPLICATION_XML)
    public Book findBook(Long id) {
        return em.find(Book.class, id);
    }
}
```



Demo



Expose entities as REST resources

JAXB and JPA integration

XML representation

Book entity

JAXB annotations

Automatic XML marshalling and
unmarshalling



Summary



Introduction

Understanding JPA

Managing Elementary Entities with JPA

Relationships and Inheritance

Querying Entities

Entity Lifecycle, Callbacks, and Listeners

Java Persistence API within Java EE



References



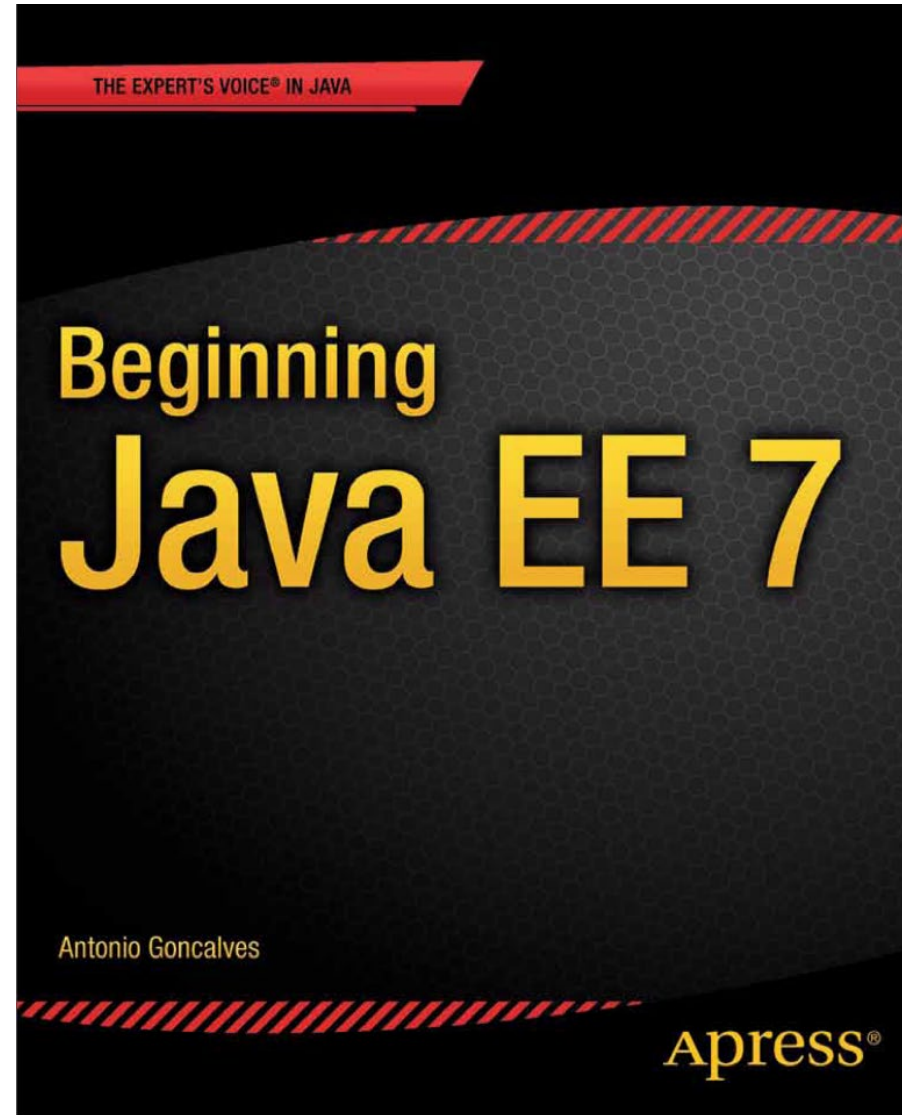
JPA 2.2 specification

- JSR 338
- <http://jcp.org/en/jsr/detail?id=338>

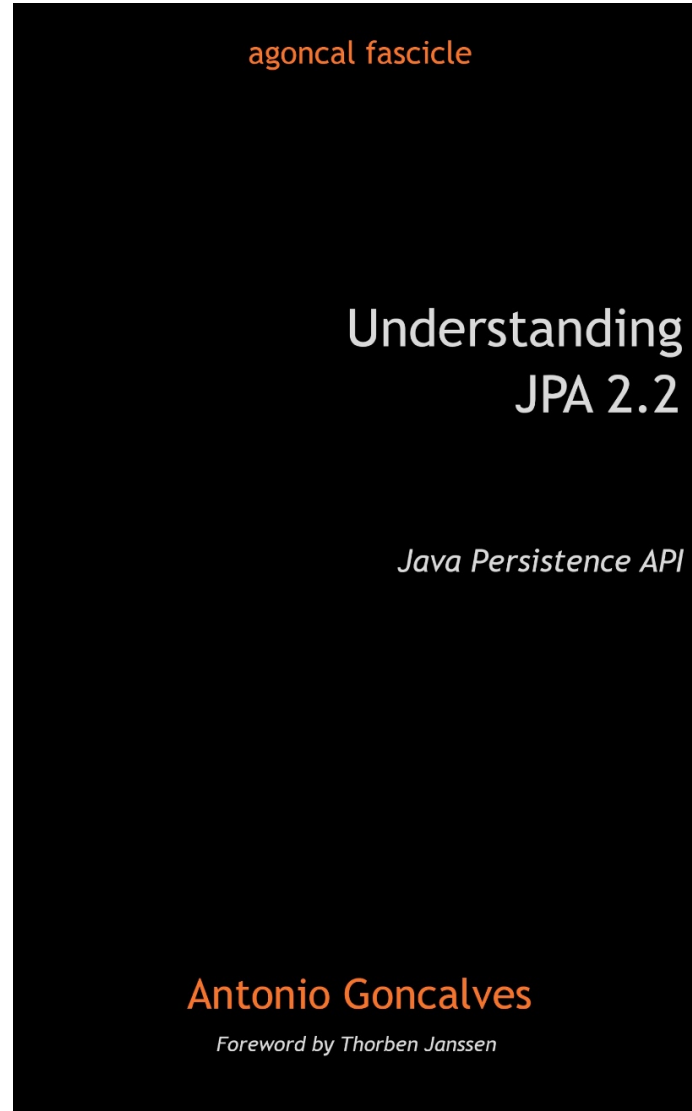
Java EE 8 specification

- JSR 366
- <http://jcp.org/en/jsr/detail?id=366>

www.amazon.com/author/agoncal



www.amazon.com/author/agoncal



Java EE Pluralsight Courses



Java Persistence API (JPA 2.2)	Bean Validation 1.1	Context and Dependency Injection (CDI 1.1)
RESTful Services in Java using Jersey	Java Web Fundamentals	Introduction to the Java API for WebSockets
Programming Servlets	Java Server Pages	Getting Started with JavaServer Faces



PLURALSIGHT

LIBRARY

Home

Browse

Paths

Channels

Bookmarks

Notes

AUTHOR TOOLS

Home

Analytics

Author's Nest

Author kit

Be a mentor

Search results for 'antonio gonalves'


All

Courses

COURSES: Video Instruction


Relevance

All Levels




Microservices: The Big Picture

antonio gonalves · Beginner · Apr 16, 2018 · 1h 45m · ★★★★★ (176)



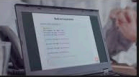
Context and Dependency Injection (CDI 1.1)

antonio gonalves · Intermediate · Mar 24, 2015 · 3h 44m · ★★★★★ (150)




Java Persistence API (JPA) 2.1

antonio gonalves · Intermediate · May 19, 2014 · 3h 43m · ★★★★★ (439)




Bean Validation 1.1

antonio gonalves · Intermediate · Jan 28, 2014 · 2h 31m · ★★★★★ (108)




Java EE: The Big Picture

antonio gonalves · Beginner · Oct 28, 2015 · 1h 9m · ★★★★★ (379)



Java EE 7 Fundamentals

antonio gonalves · Beginner · Aug 12, 2016 · 5h 25m · ★★★★★ (128)



Java EE: Getting Started

antonio gonalves · Beginner · Jun 22, 2017 · 5h · ★★★★★ (94)

antonio gonalves

antonio.gonalves@gmail.com

antonio gonalves

Java Persistence API 2.2



Antonio Goncalves

JAVA CHAMPION

@agoncal www.antoniogoncalves.org

