JSON Binding API

JSON-B

- تستخدم للتحويل من Java object إلى JSON object وبالعكس.
- تقدم مجموعة من annotations للقيام بذلك.

```java
public class Book {


    private String title;
    private String author;
    private Float price;
    private Date publishingDate;
    private int number;
    private String type;
}
```

Java Object

```json
{
    "author": "Salem",
    "number": 320,
    "price": 35.5,
    "publishingDate": "2022-05-07T07:26:34.718Z[UTC]",
    "title": "Learning Java ",
    "type": "History"
}
```

JSON Object

التحويل من Java object إلى JSON object

Create Jsonb Object

```
Book book = new Book("Learning Java ", "Salem", 35.5f, new Date(), 320, "History");
Jsonb jsonb = JsonbBuilder.create();
String result = jsonb.toJson(book);
```

Use toJson method

```
author:          "Salem"
number:          320
price:           35.5
publishingDate:  "2022-05-07T08:38:44.987Z[UTC]"
title:           "Learning Java "
type:            "History"
```

التحويل من Java object إلى JSON object

Create Jsonb Object

```
Jsonb jsonb = JsonbBuilder.create();
Book book = jsonb.fromJson("{\n"
        + "    \"author\": \"Salem\",\n"
        + "    \"number\": 320,\n"
        + "    \"price\": 35.5,\n"
        + "    \"publishingDate\": \"2022-05-07T07:26:34.718Z[UTC]\",\n"
        + "    \"title\": \"Learning Java \",\n"
        + "    \"type\": \"History\"\n"
        + "}", Book.class);
```

```
Book{title=Learning Java , author=Salem, price=35.5, publishingDate=Sat May 07 09:26:34 EET 2022, number=320, type=History}
```

Use fromJson method

Changing property name

- تستخدم @JsonbProperty لتغيير الاسم الافتراضي ل property .

```java
public class Book {


    @JsonbProperty("book-title")
    private String title;
    private String author;
    private Float price;
    private Date publishingDate;
    private int number;
    private String type;
```

```
author:          "Salem"
book-title:      "Learning Java "
number:          320
price:           35.5
publishingDate:  "2022-05-07T08:53:40.053Z[UTC]"
type:            "History"
```

Ignoring properties

- تستخدم @JsonbTransient عند الحاجة لعدم تضمين property خلال عملية التحويل.

```java
public class Book {


    @JsonbProperty("book-title")
    private String title;
    private String author;
    private Float price;
    @JsonbTransient
    private Date publishingDate;
    private int number;
    private String type;
```

```
author:       "Salem"
book-title:   "Learning Java "
number:       320
price:        35.5
type:         "History"
```

Date Formats

- تستخدم @JsonbDateFormat لتحويل التاريخ للصيغة المطلوبة.

```java
public class Book {

    private String title;
    private String author;
    private Float price;
    @JsonbDateFormat("yyyy-MM-dd")
    private Date publishingDate;
    private int number;
    private String type;
```

```
author:          "Salem"
number:          320
price:           35.5
publishingDate:  "2022-05-07"
title:           "Learning Java "
type:            "History"
```

Number Formats

- تستخدم @JsonbNumberFormat لتحويل الأرقام للصيغة المطلوبة.

```java
public class Book {

    private String title;
    private String author;
    @JsonbNumberFormat("#0.00")
    private Float price;
    private Date publishingDate;
    private int number;
    private String type;
```

```
author:          "Salem"
number:          320
price:           "35.500"
publishingDate:  "2022-05-07T09:11:33.513Z[UTC]"
title:           "Learning Java "
type:            "History"
```

Null handling

- لا يتم عمل serilalize للقيم null في JSON-B وللقيام بذلك يتم استخدام كل من:
- @JsonbNillable على مستوى class .
- @JsonbProperty على مستوى property .

```java
public class Book {

    private String title;
    @JsonbProperty(nillable=true)
    private String author;
    @JsonbNumberFormat("#0.000")
    private Float price;
    private Date publishingDate;
    private int number;
    private String type;
```

```
author:          null
number:          320
price:           "35.500"
publishingDate:  "2022-05-14T21:06:00.305Z[UTC]"
title:           "Learning Java "
type:            "History"
```

Null handling

```java
@JsonbNillable
public class Book {

    private String title;
    private String author;
    @JsonbNumberFormat("#0.000")
    private Float price;
    private Date publishingDate;
    private int number;
    private String type;
```

```
author:          null
number:          320
price:           "35.500"
publishingDate:  "2022-05-14T21:06:00.305Z[UTC]"
title:           "Learning Java "
type:            "History"
```

@JsonbPropertyOrder

- تستخدم @JsonbPropertyOrder لترتيب property في JSON .

```java
@JsonbPropertyOrder(value = {"author", "title", "price", "publishingDate", "number", "type"})
public class Book {

    private String title;
    @JsonbProperty(nillable = true)
    private String author;
    @JsonbNumberFormat("#0.000")
    private Float price;
    private Date publishingDate;
    private int number;
    private String type;
```

```
author:          "Salem"
title:           "Learning Java "
price:           "35.500"
publishingDate:  "2022-05-14T21:16:21.399Z[UTC]"
number:          320
type:            "History"
```

JsonbConfig

يمكن استخدام JsonbConfig للقيام ببعض الاعدادات قبل انشاء Jsonb Object .

```
// Create custom configuration
JsonbConfig config = new JsonbConfig();

// Create Jsonb with custom configuration
Jsonb jsonb = JsonbBuilder.create(config);
```

Naming Strategies

- تستخدم لتغيير أسماء property عند انشاء JSON .

```
JsonbConfig config = new JsonbConfig()
        .withPropertyNamingStrategy(PropertyNamingStrategy.LOWER_CASE_WITH_UNDERSCORES);
// Create Jsonb with custom configuration
Jsonb jsonb = JsonbBuilder.create(config);
String result = jsonb.toJson(book);
```

```
author:          "Salem"
title:           "Learning Java "
price:           "35.500"
publishing_date: "2022-05-15T04:46:19.469Z[UTC]"
number:          320
type:            "History"
```

```
JsonbConfig config = new JsonbConfig()
        .withPropertyNamingStrategy(PropertyNamingStrategy.UPPER_CAMEL_CASE_WITH_SPACES);
// Create Jsonb with custom configuration
Jsonb jsonb = JsonbBuilder.create(config);
String result = jsonb.toJson(book);
```

```
Author:          "Salem"
Title:           "Learning Java "
Price:           "35.500"
Publishing Date: "2022-05-15T04:47:14.036Z[UTC]"
Number:          320
Type:            "History"
```

Properties order

- لتحديد كيفية ترتيب prperties يمكن القيام بذلك من خلال PropertyOrderStrategy class .

```
JsonbConfig config = new JsonbConfig()
        .withPropertyOrderStrategy(PropertyOrderStrategy.LEXICOGRAPHICAL);
Jsonb jsonb = JsonbBuilder.create(config);
String result = jsonb.toJson(book);
```

```
author:          "Salem"
number:          320
price:           "35.500"
publishingDate:  "2022-05-15T04:55:15.554Z[UTC]"
title:           "Learning Java "
type:            "History"
```

```
JsonbConfig config = new JsonbConfig()
        .withPropertyOrderStrategy(PropertyOrderStrategy.REVERSE);
Jsonb jsonb = JsonbBuilder.create(config);
String result = jsonb.toJson(book);
```

```
type:            "History"
title:           "Learning Java "
publishingDate:  "2022-05-15T04:54:05.019Z[UTC]"
price:           "35.500"
number:          320
author:          "Salem"
```

Null handling

- يمكن تحديد التعامل مع null value لكل properties من خلال استخدام JsonbConfg .

```
JsonbConfig config = new JsonbConfig()
        .withNullValues(true);
Jsonb jsonb = JsonbBuilder.create(config);
String result = jsonb.toJson(book);
```

```
author:          null
number:          320
price:           "35.500"
publishingDate:  "2022-05-15T07:14:28.319Z[UTC]"
title:           "Learning Java "
type:            "History"
```

Adapter

- Adapter عبارة عن class تستخدم في JsonbAdapter interfcae لعمل serialize/deserialize بالطريقة التي نرغب.

```
Jsonb jsonb = JsonbBuilder.create();
String result = jsonb.toJson(book);
```

```
author:          "Salem"
number:          320
price:           "35.500"
publishingDate:  "2022-05-15T05:04:28.88Z[UTC]"
title:           "Learning Java "
type:            "History"
```

```
JsonbConfig config = new JsonbConfig()
        .withAdapters(new BookAdapter());
// Create Jsonb with custom configuration
Jsonb jsonb = JsonbBuilder.create(config);
String result = jsonb.toJson(book);
```

```
Author:    "Salem"
title:     "Learning Java "
```

Adapter Class

```
public class BookAdapter implements JsonbAdapter<Book,JsonObject> {

    @Override
    public JsonObject adaptToJson(Book book) throws Exception {
        return Json.createObjectBuilder()
            .add("Author", book.getAuthor())
            .add("title", book.getTitle())
            .build();
    }

    @Override
    public Book adaptFromJson(JsonObject jsonObject) throws Exception {
        Book book = new Book();
        book.setAuthor(jsonObject.getString("author"));
        book.setTitle(jsonObject.getString("title"));
        return book;
    }

}
```