**SMVC-372-Jackson:  
  
Showcase delle annotations Jackson:**

**AFU:**

**Bean Annotations:**

1. @JsonProperty
2. @JsonPropertyOrder
3. @JsonGetter
4. @JsonAnyGetter (generica Map per le properties)
5. @JsonSetter
6. @ JsonAnySetter
7. @JsonRawValue (skipParsing for that field)
8. @JsonRootName
9. @JsonIgnoreProperties
10. @JsonIgnore
    1. In caso di @JsonView questo ultimo ha la precedenza.
11. @JsonIgnoreType (nested classes)
12. @JsonInclude (gestione null e defaults)
13. @JsonFormat (format data)
14. @JsonView
15. @JsonTypeInfo (Ereditarieta)
16. @JsonSubTypes (Ereditarieta)
17. @JsonFilter

**Demo:**

Lanciare UTests dopo modifiche a classi Dto annotate con Jackson.  
Eseguire Utest uno alla volta.

**Test Serializzazione/Deserializzazione Dto:**Seguire UTests e abilitare/disabilitare commenti su Contact

**TC1** Campi serializzati e campi ignorati  
@JsonIgnore singolo  
@JsonIgnoreProperties  
@JsonPropertyOrder  
custom serialization Date (vedi UTests)  
Nel momento in cui implement un CustomSerializer dovro’ probabilmente anche implementare un CustomDeserializer.  
  
**TC2) Serializzazione/Deserializzazione Map<K,Dto>**  
Serializzazione semplice. Map -> Json  
Deserializzazione:

- Quale Map impl?  
- Come Deserializzare in generics? ->

Class abstract TypeReference.  
Istanze create con classi anonime (non necessaria classe di estensione).  
Passare oggetto wrapper TypeReference nel readValue.

**TC3) Set<T>**java 2 json -> array OK  
json 2 java -> Quale collection? -> TypeReference

**TEST:**

1. **Serializzazione/Deserializzazione**
2. **Ignore (static)**
3. **Filters (dynamic ignore)**
4. **Views**
5. **Serializzazione/deserializzazione streaming:**

**DESIGN:**

**Relazioni**- Bidirezionali  
- Ereditarieta  
- Filtri sui campi (Ignore dinamico).   
Scelgo dinamicamente quali campi serializzare.

**Ereditarieta:**Annotatare superclasse con @JsonTypeInfo e @JsonSubTypes.  
Definire un campo discriminante per identificare subclass.   
Definire lista di possibili Subclasses.  
Costruttore default deve inizializzare il campo discriminante.

**Bidirezionali**JsonView.   
Scelgo in base al metodo da quale direzione serializzare la relazione.

**Spiegazioni:**

1. **UTest01: Serializzazione/Deserializzazione  
     
   Date:**Serializzazione di default mette date in timestamp.  
   Possibile customizzare questo comportamento con
   1. annotation @JsonFormat
   2. custom serializer @JsonSerialize

**Collections**:  
- java 2 json -> js array  
- js array -> Collection<T> -> usare TypeReference (Jackson)

**Properties**: Controllo  
- name prop serializzata  
- order

1. **Ignore (static)**

Campi da ignorare in ser/des

1. **Filters:**  
   Problema: decidere dinamicamente quali campi includere e quali ignorare.  
   Soluzioni:
   1. Restituire Map invece che oggetto.  
      P: Design.  
      Scomoda per chiamate a service (che usano bean)
   2. Marcare I campi nullable come campi da ignorare @JsonInclude(Include.NON\_NULL)  
      Si propaga fino a Dao. Cercare solo campi da popolare e mettere gli altri a null.
   3. Filters. Annotation in bean e FilerProvider aggiunti al ObjectMapper.  
      P: Performance.  
      Bisogna creare un writer per ogni richiesta. Operazione costosa.

1. **Views:**Decidere di mostrare un solo livello (o direzione) di una relazione tra beans.  
   - Impostare un livello base (EntityData) commune a tutti.  
   - Estendere il livello base con Details, che vengono utilizzati solo in particolari view.
2. **Serializzazione/deserializzazione streaming:**

**RESOURCES:  
Archivi/Edizioni/2017/SMVC/Resources/Jackson**