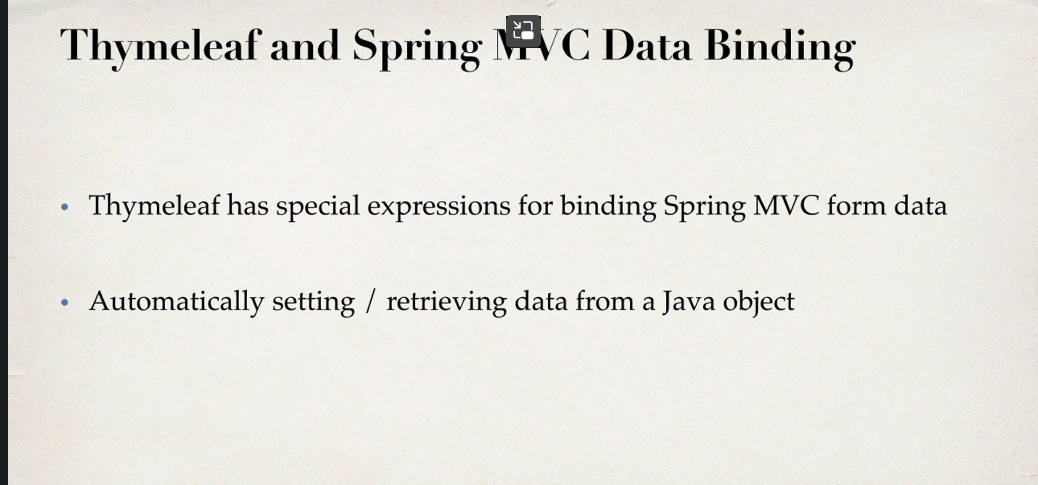
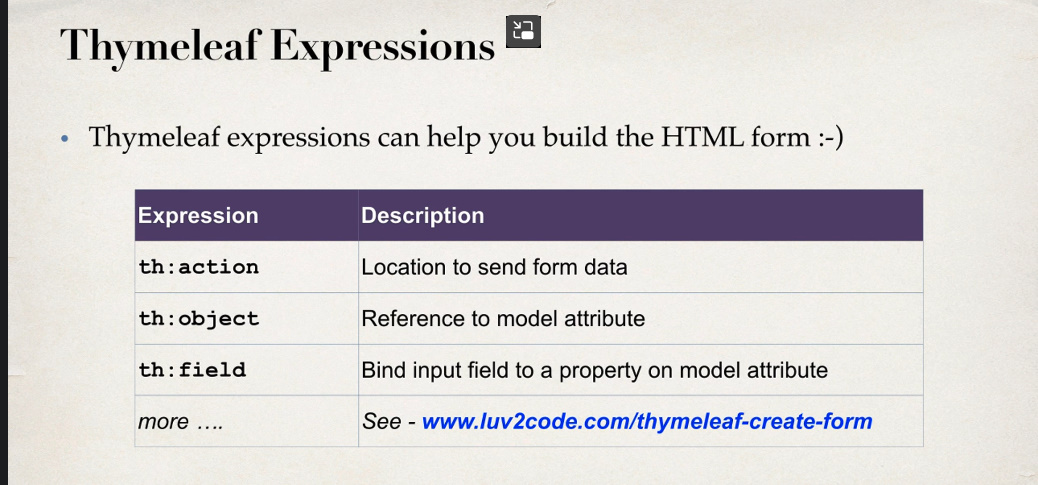
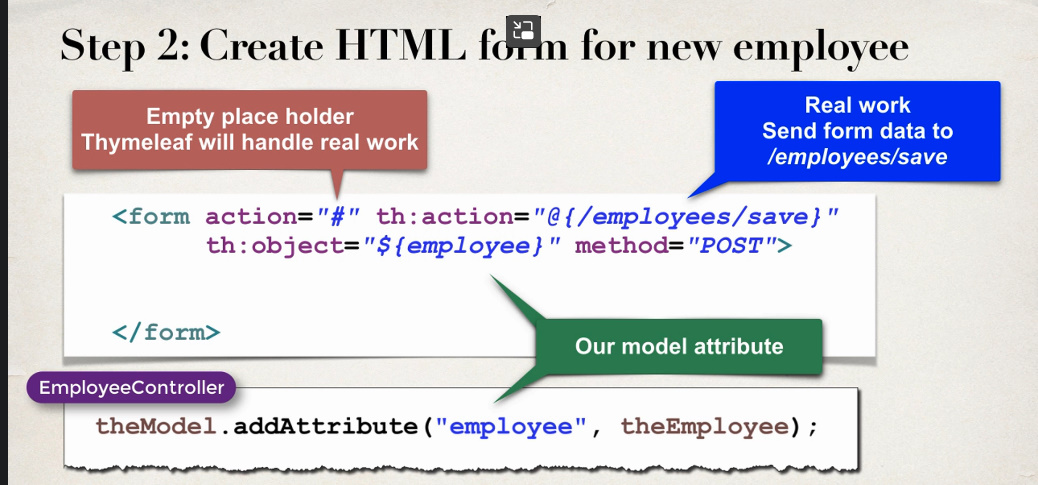
* Cand foloseam Hibernate pur, noi extrageam un obiect cu session.get(id) si cand returnam acel obiect, daca il modificam pana sa se dea commit la sesiune, fara a folosi vreo alta metoda sau query, el era actualizat automat si in baza de date. Hibernate returneaza referinte la obiectele extrase din baza de date, si cand se termina sesiunea, dupa commit, le salveaza iar in baza de date ca poate am modificat noi ceva. **Acelasi principiu e valabail si in JPA si Spring Data**
* Avem grija la metoda save(),din JPA, care creaza un nou obiect copie al cerui dat, dar i va seta un id, si va returna obiectul noua dupa ce il pune in DB, nu va lua direct obiectul nostru si sa puna in el id, ca Hibernate.
* 
* 
* 

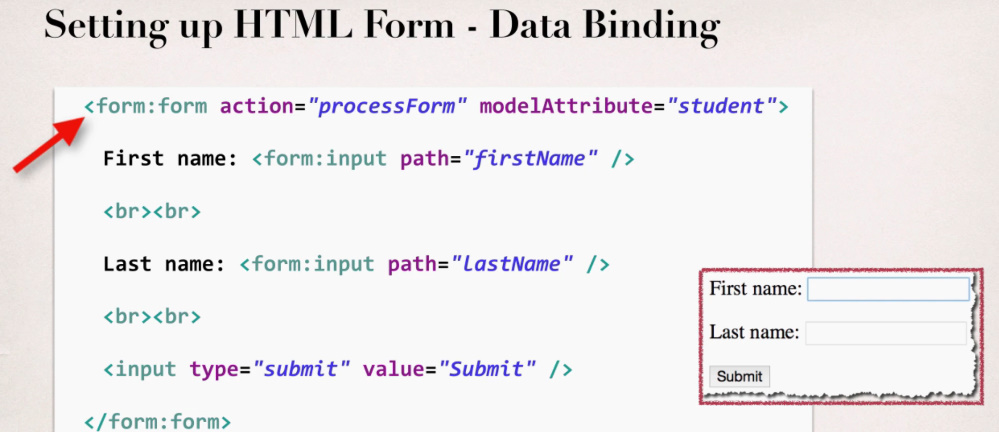
JSP avea un form:form tag, thymeleaf nu are, dar foloseste th:

* 

exista si th:method, dar nu e nevoie de el

action=”” e necesara la orice forma, dar punem “#” ca sa nu faca nimic

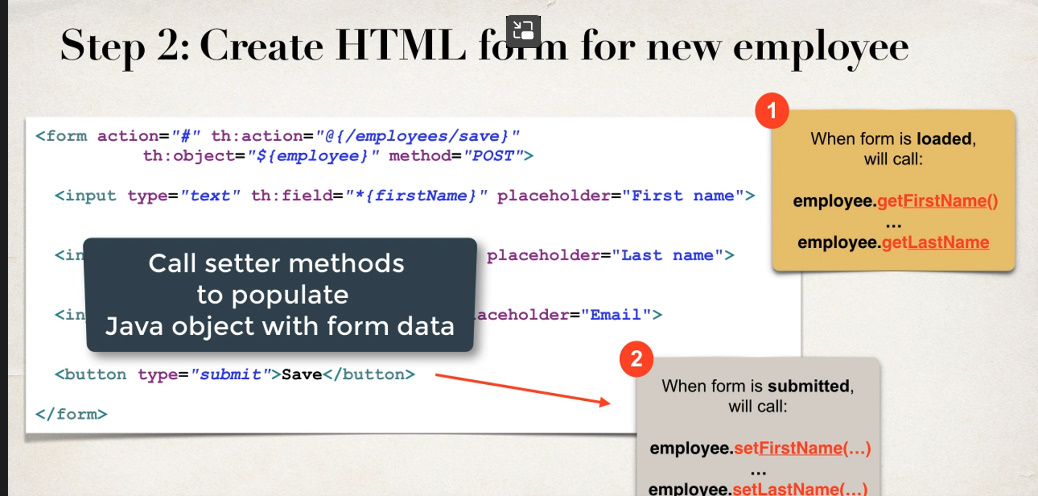
@{} – este context path, gen daca aplicatia noastra are context path myapp, atunci orice link mereu va incepe cu localhost:8080/myapp, si anume acest @{} se va asigura ca /myapp/ sa fie mereu pus automat.

* 

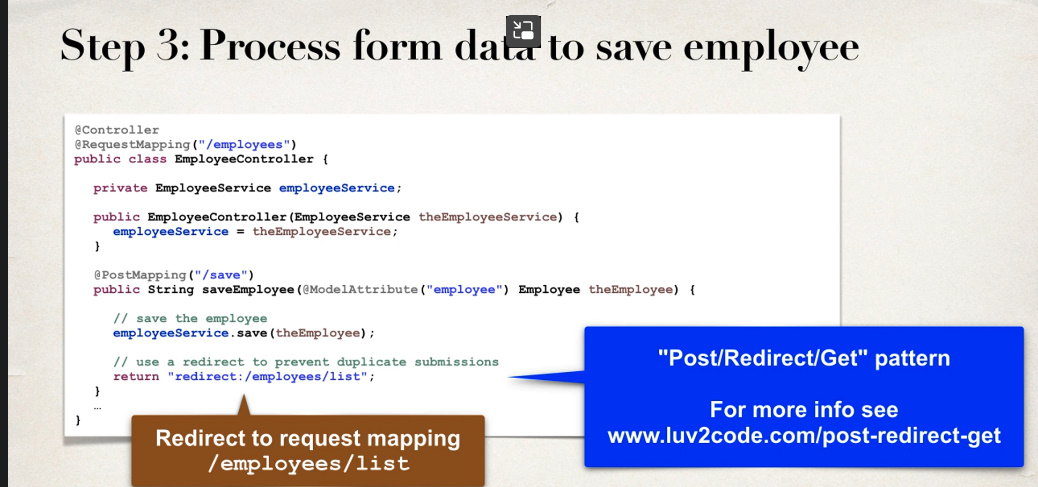


Anume \* face referinta la obiectul luat din Model, adica din th:object

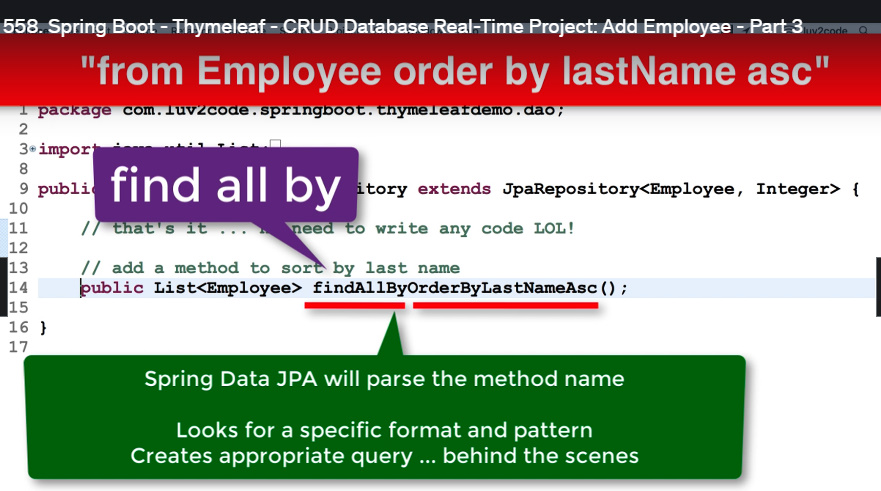
* In JSP punem modelAttribute=”nume”, in th punem th:object=”${}”
* In JSP punem path=”var” in th punem th:field=”\*{var}”
* La fel ca modelAttribute, th:object se asigura sa creeze un nou object de tipul celui trimis, sa il umple cu date si sa il adauge in Model al paginii ce se duce la Controller cu acelasi nume.



Deci intai se vor chema metodele get pentru a pune obiectele in forma si apoi set pentru a popula noul obiect creat pe baza datelor din forma.



**Add custom method on JPA Repository**

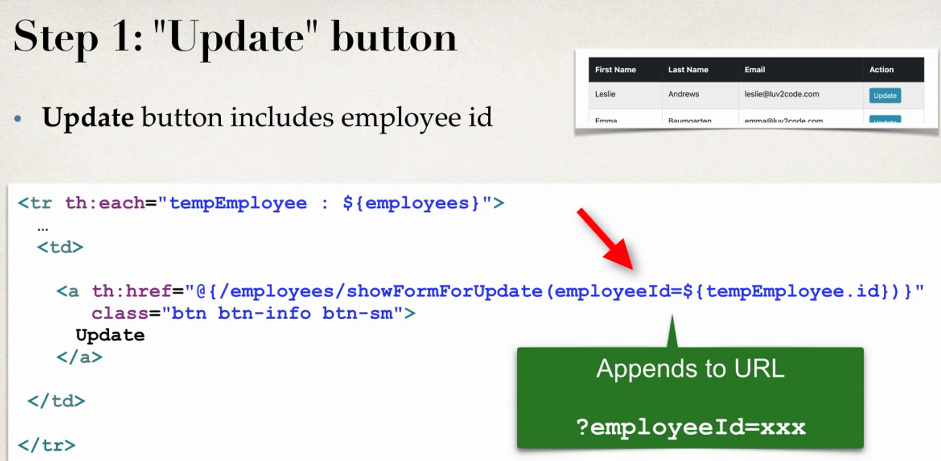
* Putem adauga propriile noastre metode in interfata care extinde JpaRepository
* 
* Deci, Spring Data poate crea query pe baza la numele metodei dat de noi. Odata ce dupa findAll am scris By, se va incepe crearea la QUERY, care va fi asa cum vom scrie noi numele metodei, de ex:

OrderByLastNameAsc = order by lastName asc

[Spring Data JPA - Reference Documentation](https://docs.spring.io/spring-data/jpa/docs/current/reference/html/#repositories.query-methods.details)

**Update Employee**

* Avem nevoie ca la apasarea butonului update sa se trimita un param ce contine id al employee. In JSP trebuia sa cream noi un request ce sa fie legat de un param cu date setate de noi(proprietati)
* Thymeleaf aici ne usureaza munca. Putem noi direct sa cream acel param cu property ca id in link sau request trimis:



pentru a adauga un nou request in cel prezent deja, folosim (NumeParam=${attribute.field})

