How to Catch Collections in Form Application By Using @InitBinder Annotation

`@InitBinder` annotations is used to resolve type mismatch and bind exceptions occured in a form application.

In this tutorial, we will try to explain how to use `@InitBinder` annotation to catch submitted collections in a form application.

Lets create a JSP file as follows:

```
<form:form commandName="belge" action="/kullanici/belge/kaydet">
    <form:select multiple="true" path="belgeSatirlar[0].gumrukKodlari">
        <c:forEach items="${gumrukler}" var="gumruk">
            <option value="${gumruk.kod}">${gumruk.aciklama}</option>
        </c:forEach>
    </form:select>
    <form:select multiple="true" path="belgeSatirlar[1].gumrukKodlari">
        <c:forEach items="${gumrukler}" var="gumruk">
            <option value="${gumruk.kod}">${gumruk.aciklama}</option>
        </c:forEach>
    </form:select>
</form:form>
```

As seen, there is a form having command **"belge"** and this entity contains **belgeSatirlar** field and **belgeSatirlar** contains **gumrukKodlari.** We can select multiple **gumrukKod** thanks to **multiple** attribute of the **select** element. Note that there are two select element in the form, first select element's path attribute value is "belgeSatirlar[0].gumrukKodlari" and the other's value "belgeSatirlar[1].gumrukKodlari". After submitting the form, you will get an exception: "Rejected value **belgeSatirlar[0].gumrukKodlari...".** To solve this problem we will use `@InitBinder` annotation, but lets look at **Belge**, **BelgeSatir** and **GumrukKod** entities.

The definition of **Belge** entity as follows:

```
@Entity
@Table(name = "BelgeKayit")
public class Belge {
     @Id
     @GeneratedValue(strategy = GenerationType.IDENTITY)
     @Column(name = "BelgeKayitId")
     private int belgeId;
     @OneToMany(mappedBy = "belge",cascade = CascadeType.ALL,fetch = FetchType.EAGER)
     private List<BelgeSatir> belgeSatirlar;
}
```

And the definition of **BelgeSatir** as follows:

```
@Entity
@Table(name = "BelgeSatirKayit")
public class BelgeSatir {
     @Id
     @GeneratedValue(strategy = GenerationType.IDENTITY)
     @Column(name = "BelgeSatirId")
     private int belgeSatirId;
     @ManyToMany(fetch = FetchType.EAGER)
     @JoinTable(
     name="BelgeSatir_GumrukKod",
     joinColumns = @JoinColumn( name="BasvuruNo"),
     inverseJoinColumns = @JoinColumn( name="GumrukKod")
     )
     private List<GumrukKod> gumrukKodlari;
}
```

And the definition of **GumrukKod** as follows:

```
@Entity
@Table(name="GumrukKodu")
@org.hibernate.annotations.Cache(usage = CacheConcurrencyStrategy.READ_ONLY)
public class GumrukKod {
```

```
5         @Id
6         @Column(name="KOD")
7         private String kod;
8
9         @Column(name="Aciklama")
10        private String aciklama;
11    }
```

Controller Class

```
1   @Controller
2   @RequestMapping(value = "/kullanici", method = RequestMethod.GET)
3   public class UserController {
4         @Autowired
5         private GumrukService gumrukService;
6         @InitBinder
7         protected void initBinder(WebDataBinder binder) throws Exception {
8             MyWebDataBinder myWebDataBinder=new MyWebDataBinder(binder);
9             myWebDataBinder.registerGumrukCollectionEditor(List.class,
10                                                  "belgeSatirlar.gumrukKodlari",gumrukService);
11        }
12  }
```

After configuration specified at line Spring converts **belgeSatirlar[0].gumrukKodlari** and **belgeSatirlar[1].gumrukKodlari** into List collections

MyWebDataBinder Class

```
import org.springframework.beans.propertyeditors.CustomCollectionEditor;
import org.springframework.web.bind.WebDataBinder;
import java.util.Collection;
import java.util.List;

public class MyWebDataBinder {
     private WebDataBinder webDataBinder;
     public MyWebDataBinder(WebDataBinder webDataBinder) {
         this.webDataBinder=webDataBinder;
     }

     public void registerGumrukCollectionEditor(Class<?> requiredType, String field, GumrukService
gumrukService) {
         webDataBinder.registerCustomEditor(requiredType, field, new
GumrukCollectionEditor(List.class, gumrukService));
     }

     private class GumrukCollectionEditor extends CustomCollectionEditor {
         private GumrukService gumrukService;

         public GumrukCollectionEditor(Class<? extends Collection> collectionType, GumrukService
gumrukService) {
             super(collectionType);
             this.gumrukService= gumrukService;
         }

         protected Object convertElement(Object element) {
             if (element instanceof GumrukKod) {
                 return element;
             }
             if (element instanceof String) {
                 GumrukKod gumrukKod = gumrukService.get((String) element);
                 return gumrukKod;
             }
```

```
                return null;
            }
        }
    }
```