Développement avancé d'application Web

Informatique de gestion – Bloc 3

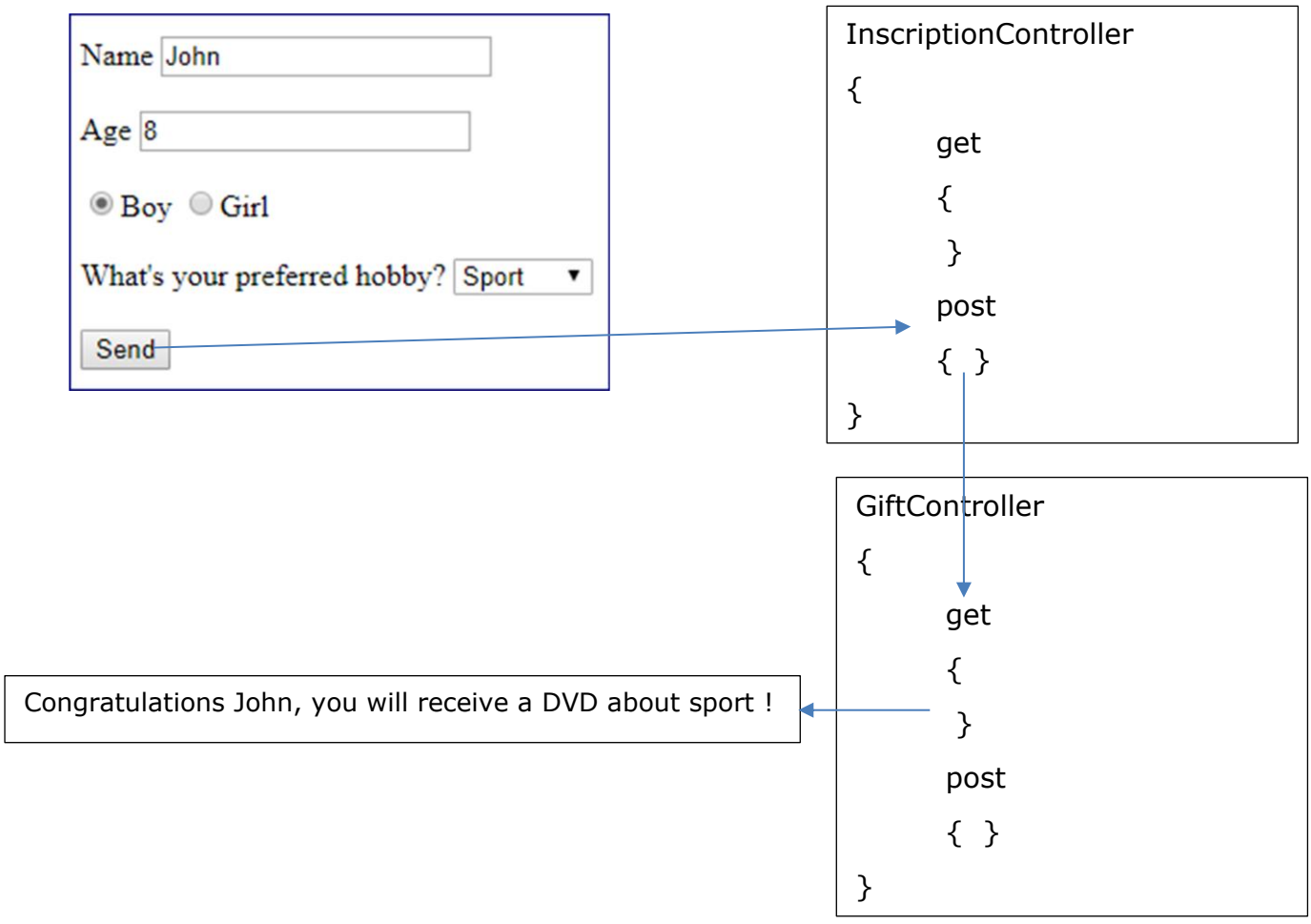
Haute-École de Namur-Liège-Luxembourg

Labo Spring 4
- *Attribut de session* -

Objectif
- Définir et utiliser des attributs liés à la session de l'utilisateur ainsi qu'un service

Dans la suite de l'exercice, en fonction de l'âge et du hobby de l'enfant, un cadeau va lui être attribué. Le cadeau attribué sera affiché dans la page gift.jsp.

Le calcul du cadeau va se faire dans la méthode *get* du controller associé à cette page, soit la classe *GiftController*. La méthode *post* de *InscriptionController* devra faire une redirection vers *GiftController*.



Etape 1 – L'objet de type *User* placé en session

Portée des données placées dans le dictionnaire Model

Les données placées dans le dictionnaire *Model* n'ont qu'une durée de vie limitée, juste le temps de la requête (***Request Scope***), c'est-à-dire jusqu'à ce que la réponse à la requête (la page) soit envoyée au client.

Pour que la classe *GiftController* puisse calculer le cadeau de l'enfant, il faut pouvoir récupérer les informations encodées par l'enfant et récupérées dans l'objet de type *User* via la méthode *post* de *InscriptionController* (cet objet a été annoté *@ModelAttribute*). Or, cet objet n'existe que le temps de la requête (jusqu'à l'affichage de la page) ; il n'est donc pas accessible dans un autre controller.

Pour que cet objet persiste durant toute la session de l'utilisateur, il faut le déclarer attribut de session dès sa création dans *InscriptionController*.

Attribut en session

Certaines données doivent être accessibles par toutes les pages d'une même session d'utilisateur. On déclare ces données comme attributs placés en session ; l'état de ces données est persisté le temps de la session de l'utilisateur (***Session Scope***), c'est-à-dire entre les différentes requêtes de la session. Les attributs placés en session sont donc accessibles par toutes les pages.

Un attribut est déclaré en session via les annotations *@SessionAttributes* et *@ModelAttribute*.

Spring créera un nouvel objet par session d'utilisateur (inversion de contrôle) et gérera ces objets placés en session.

Pour que les données de l'enfant stockées dans l'objet *User* soient accessibles par toutes les pages d'une même session d'utilisateur, cet objet *User* va être placé en session ; il sera accessible durant toute la session de l'utilisateur.

Le nom de l'attribut à placer en session va intervenir à plusieurs endroits dans le code. Le mieux est de déclarer une **constante** pour ce nom (prévoir par exemple une classe reprenant toutes les constantes publiques).

```
public class Constants {
    public static final String CURRENT_USER = "currentUser";
}
```

Il faut créer et initialiser l'attribut à placer en session. Ceci va se faire via une méthode dans le controller (peu importe le nom de la méthode) dont le type de retour correspond au type de l'attribut et le corps de la méthode comprend les instructions d'initialisation de l'attribut. Cette méthode est annotée *@ModelAttribute :* utilisez la constante comme valeur de l'annotation (cette valeur correspondra au nom de l'attribut ainsi créé et initialisé par la méthode).

Pour préciser qu'un attribut est de type session, annotez @*SessionAttributes* la classe controller en utilisant de nouveau la constante comme valeur de l'annotation.

```
import org.springframework.web.bind.annotation.SessionAttributes;
import org.springframework.web.bind.annotation.ModelAttribute;

@Controller
@RequestMapping(value="/inscription")
@SessionAttributes({Constants.CURRENT_USER})
public class InscriptionController {

    @ModelAttribute(Constants.CURRENT_USER)
    public User user()
    {
        return new User();
    }
```

Restez cohérent en utilisant toujours le même nom pour cet attribut (via appel à la constante), y compris dans sa déclaration comme argument de la méthode *post*.

```
@RequestMapping(value="/send", method=RequestMethod.POST)
public String getFormData(Model model,
                          @Valid @ModelAttribute(value=Constants.CURRENT_USER) User user,
                          final BindingResult errors) {
```

Dans la page *userInscription.jsp,* veillez à utiliser comme valeur du paramètre *modelAttribute* du formulaire la même valeur que celle stockée dans la constante.

```
<form:form  id="inscription"
            method="POST"
            action="/firstSpring/inscription/send"
            modelAttribute="currentUser">
```

Etape 2 – Service *GiftService*

Le cadeau à attribuer à l'enfant doit être calculé par une classe de type service sur base de l'âge de l'enfant et de son hobby.

Dans le package *service*, créez la classe *GiftService* qui contient une méthode publique *chooseGift* qui reçoit en argument un hobby (*String*) et un âge (*int*) et qui retourne le cadeau (*String*). Cette méthode attribue un cadeau à l'enfant sur base des règles suivantes (données à titre d'exemple) :

Avant 5 ans : un puzzle concernant son hobby
Entre 5 et 10 ans : un DVD concernant son hobby
Après 10 ans : un livre concernant son hobby

Cette classe doit être annotée *@Service*. Ceci permettra d'y avoir accès par ***injection de dépendance***.

```
import org.springframework.stereotype.Service;

@Service
public class GiftService {
```

Etape 3 – Classe *GiftController*

Créez la classe *GiftController* dans le package *controller*.

Récupérez dans cette classe la référence vers l'utilisateur courant placé dans la session, et ce, en utilisant les annotations *@SessionAttribute* sur la classe controller et *@ModelAttribute* sur l'argument de la méthode *get*.

```
@Controller
@RequestMapping(value="/gift")
@SessionAttributes({Constants.CURRENT_USER})
public class GiftController {
```

```
@RequestMapping(method=RequestMethod.GET)
public String home(Model model,
                   @ModelAttribute(value=Constants.CURRENT_USER) User user) {
```

Dans la classe *GiftController*, récupérez par injection de dépendance une référence vers un objet de la classe *GiftService*. Pour l'utilisation d'un service par injection, référez-vous aux étapes 8 et 9 du labo 3.

Dans la méthode *get*, appelez sur cet objet de type *GiftService* la méthode *chooseGift* en lui passant l'âge et le hobby de l'enfant. Ajoutez le cadeau ainsi calculé comme attribut dans le *Model*, afin de pouvoir l'afficher dans la page *gift.jsp*.

Faites l'étape 4 avant de tester votre application !

Etape 4 – Classe *InscriptionController*

N’oubliez pas de modifier la méthode *post* de *InscriptionController* afin de rediriger vers la méthode *get* du controller *GiftController*.