



HTTP verbs

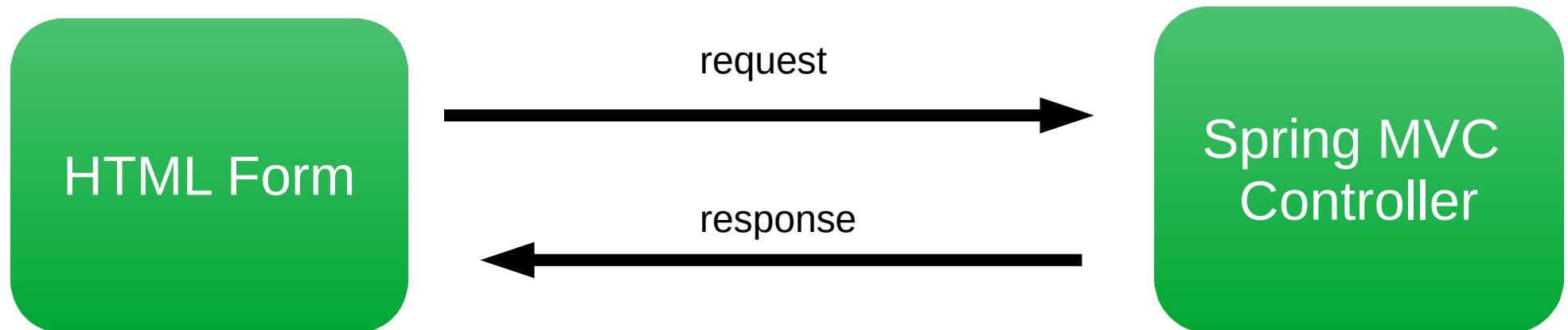
@GetMapping & @PostMapping ...

VS

RequestMapping

Controller refactoring

- Nouvelles annotations dans spring
- @GetMapping
- @PostMapping



Envoyer des données avec GET


```
<form action="processForm" method="GET">  
  <input path="prenom" />  
  <input path="nom" />  
  ...  
  <input type="submit" value="Valider" />  
</form>
```

Quand on envoie des données en GET , les informations sont passées dans la query String , sous forme de paires clé=valeurs séparées par des &.

Ici dans l'exemple ci dessus , on obtient : urlAdresse?
prenom=valeur&nom=valeur...

Soumettre un formulaire

Cette annotation est générale
Elle mappe tous les types de requête.




```
@RequestMapping("/processForm")
public String processForm( ... ){
    ...
}
```

Avec cette dernière syntaxe, on peut
réduire , limiter le champ des requêtes
concernées par ce mapping.
Seules les requête adressées en GET
seront traitées par la méthode
processForm().

```
@RequestMapping(path="/processForm",
method=RequestMethod.GET)
public String processForm( ... ){
    ...
}
```

Nouvelle syntaxe



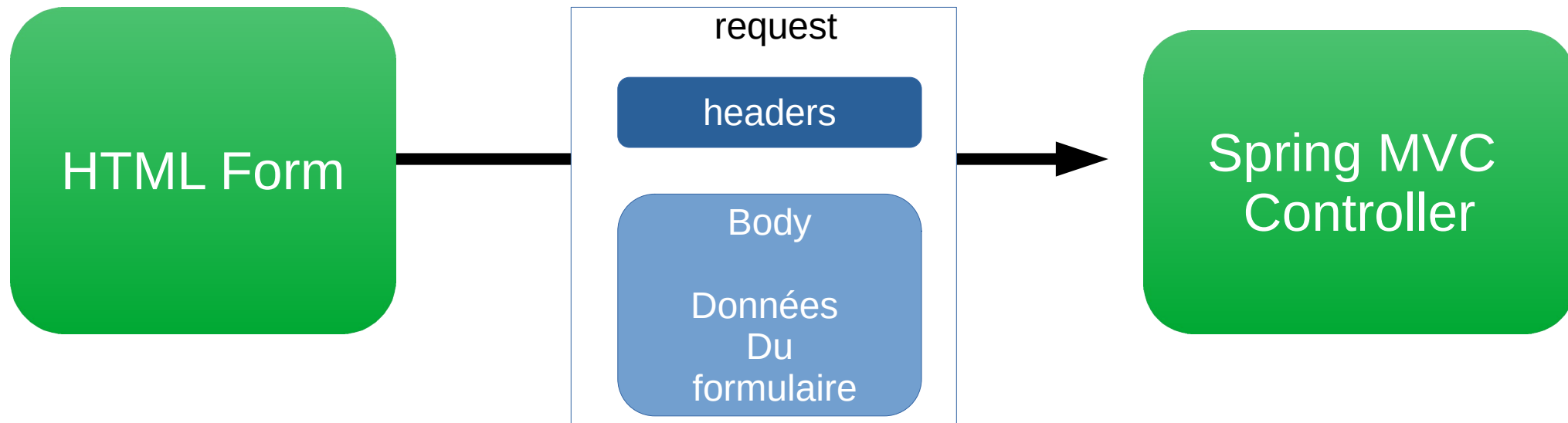
```
@GetMapping("/processForm")
public String processForm( ... ){
    ...
}
```

HTTP Post :

```
<form action="processForm" method="POST">  
  <input path="prenom" />  
  ...  
  <input type="submit" value="Valider" />  
</form>
```


Avec la méthode POST, les données ne sont plus visibles dans l'adresse.
Ici les données seront passées au contrôleur dans le corps de la requête HTTP.

On peut illustrer l'exemple ci-dessus par le schéma suivant :



@PostMapping

Cette annotation est générale
Elle mappe tous les types de requête.




```
@RequestMapping("/processForm")
public String processForm( ... ){
    ...
}
```

Cette précision va mapper uniquement les
requêtes transmises en POST

```
@RequestMapping(path="/processForm",
    method=RequestMethod.POST)
public String processForm( ... ){
    ...
}
```

Cette annotation va mapper uniquement les
requêtes transmises en POST



```
@PostMapping("/processForm")
public String processForm( ... ){
    ...
}
```

Quel critère pour choisir entre GET et POST ?

GET

Débug



Favoris et envoi par email de lien



Limitée en taille des données

Données exposées

POST

Pas de limitation en taille des données



Données non exposées



Données binaires



À vous de choisir quand vous utilisez une méthode HTTP ou l'autre.

Refactoring Controller

```
@Controller
@RequestMapping("/user")
public class UserController {

    @Autowired
    private UserDao userDao;

    @RequestMapping("/list")
    public String listCustomers(Model model) {
        List<User> users = userDao.getUsers();
        model.addAttribute("users", users);
        return "list-users";
    }
}
```

```
@Controller
@RequestMapping("/user")
public class UserController {

    @Autowired
    private UserDao userDao;

    @GetMapping("/list")
    public String listCustomers(Model model) {
        List<User> users = userDao.getUsers();
        model.addAttribute("users", users);
        return "list-users";
    }
}
```


Refactoring Controller

```
@Controller
@RequestMapping("/user")
public class UserController {

    @Autowired
    private UserDao userDao;

    @RequestMapping("/list")
    public String listCustomers(Model model) {
        List<User> users = userDao.getUsers();
        model.addAttribute("users", users);
        return "list-users";
    }
}
```

```
@Controller
@RequestMapping("/user")
public class UserController {

    @Autowired
    private UserDao userDao;

    @GetMapping("/list")
    public String listCustomers(Model model) {
        List<User> users = userDao.getUsers();
        model.addAttribute("users", users);
        return "list-users";
    }
}
```