**Get vs Post**

**Get** – preia date de la reursa data

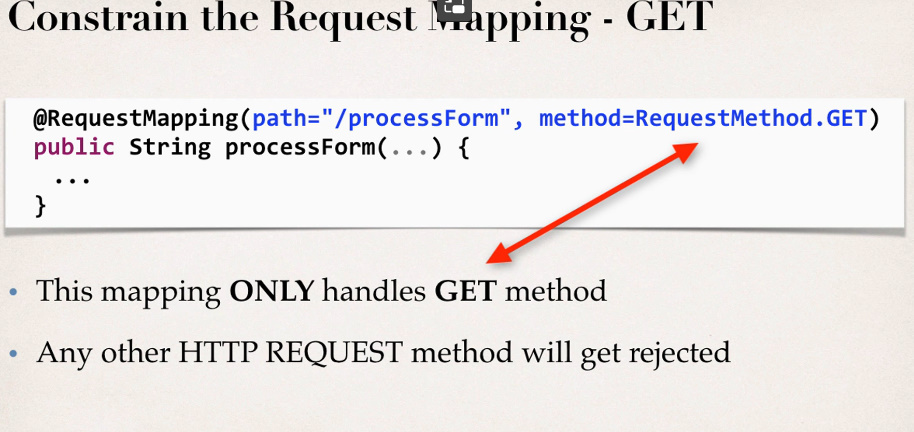
**Post** – trimite date catre resursa data

**Despre Forma**

* Fie ca forma foloseste metoda GET sau POST, acesta metoda va fi folosita doar cand se apasa pe un buton! Forma nu poate singura face vreun request.

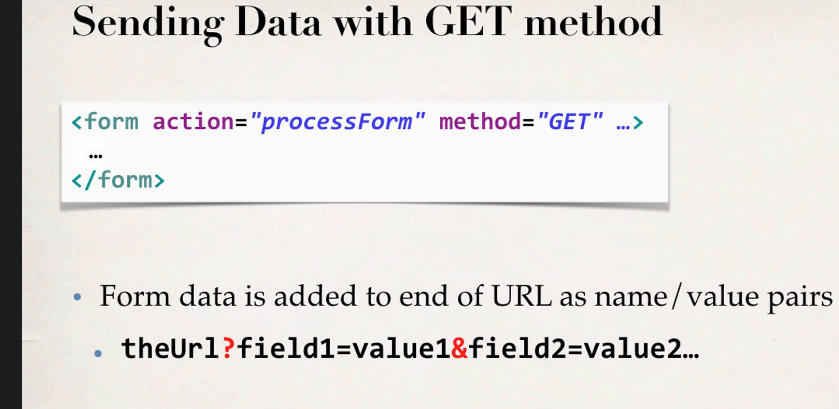
**@RequestMapping**

* Se ocupa de toate metodele HTTP, adica GET,POST, PUT etc.
* Putem specifica insa de care metoda anume sa se ocupe asa:



**@GetMapping**

* La GET,Datele de la forma sunt **adaugate doar** la final de URL cu perechi nume : valoare, **deci nu poate trimite atribute, doar parametri!**

****

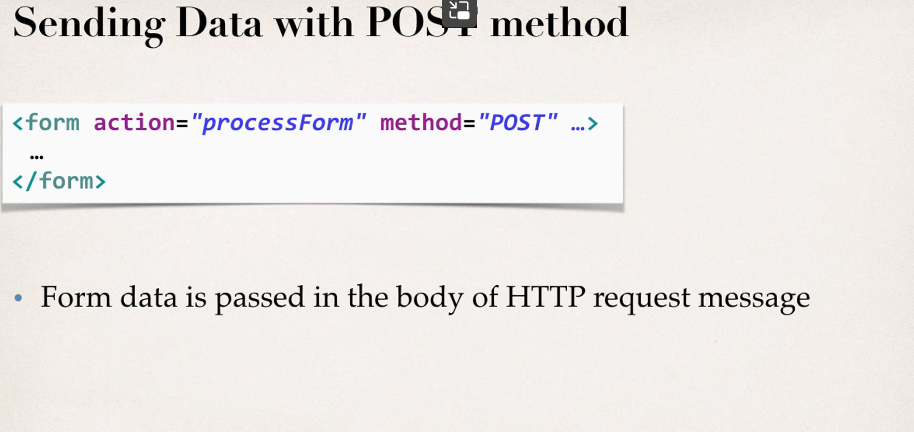
* @**GetMapping** este prescurtarea lui @RequestMapping(path=”” method=”RequestMethod.GET). Asta inseamna ca metoda va raspunde doar in cazul in care HTTP Request trimis este GET. Adica, asta inseamna ca forma are method=”GET” si deci are nevoie ca Controllerul sa trimita date, iar @GetMapping spune ca metoda e pentru forma care are method=”GET”, adica are nevoie de date, si deci **metoda va trimite date la forma**. Nu inseamna ca metoda va obtine date trimise de forma, caci forma nu trimite nimic daca are method=”GET”
* Este doar la nivel de metoda

@Autowired  
 private CustomerDAO customerDAO;  
 @GetMapping("/list")  
 public String listCustomers(Model model){  
List<Customer> customerList = customerDAO.getCustomers();  
model.addAttribute("customers",customerList);  
 return "list-customers";  
 }

Deci, cand forma are methd=”GET”, aceasta metoda se va ocupa de request si va trimite date, asa cum forma cu GET method are nevoie de date.

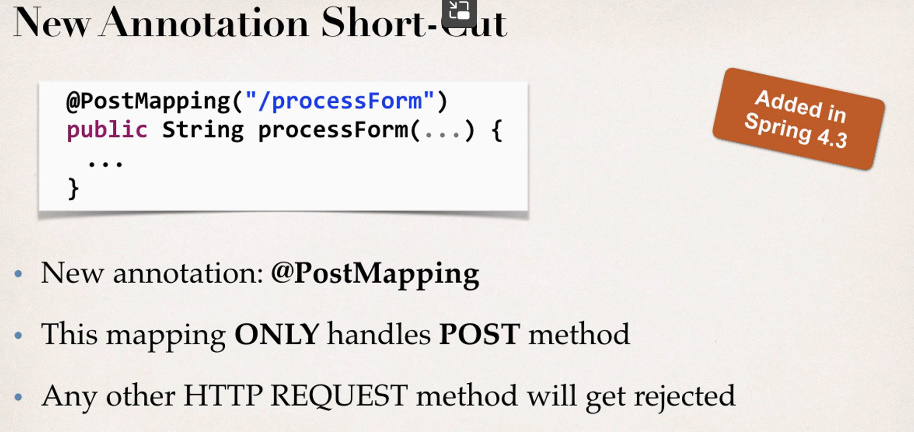
**@PostMapping**

* POST trimite date



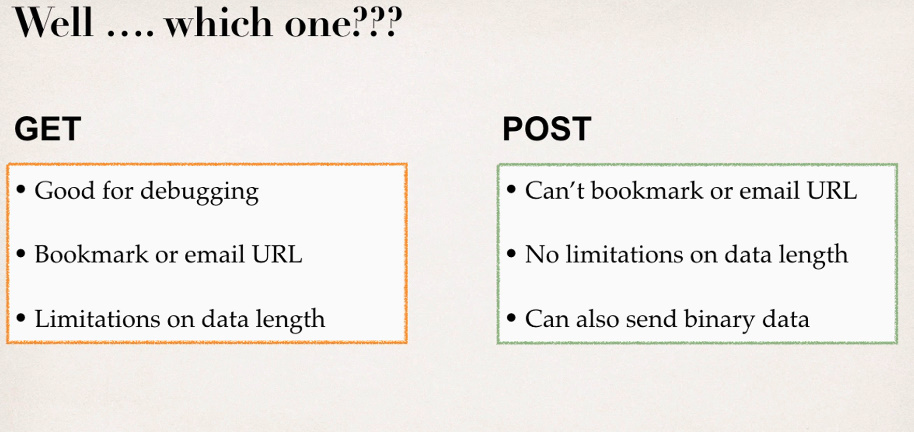


* **POST nu trimite date in url la request! Ci doar ca flux de biti, dar poate trimite param oricum in url.**
* @**PostMapping** este prescurtarea lui @RequestMapping(path=”” method=”RequestMethod.POST). Asta inseamna ca metoda va raspunde doar in cazul in care HTTP Request trimis este POST. Adica, asta inseamna ca forma are method=” POST” si deci inseamna ca va trimite un request cu date catre Controller, iar @PostMapping spune ca metoda e pentru forma care are method=”POST”, adica forma vrea sa trimita date,nu sa obtina, si deci **metoda va obtine date de la forma**. Nu inseamna ca metoda va trimite date la forma, caci forma nu are nevoie de date daca are method=”POST”.
* Totusi, forma poate lua date de la un atribut, daca pagina deschisa a fost deschisa de un request cu GET, si deci are atribute in ea
* Este doar la nivel de metoda



* Chiar daca folosim post, nu inseamna caa metoda ce se ocupa de el nu poate returna tot aceeasi pagina. Nu are importanta.

**Get vs Post**



Totusi, get si post in multe cazuri pot fi folosite ambele, caci ele fac cam acelasi lucru, insa modul cum o fac difera.

De ex, daca avem o pagina cu o lista de useri si avem un camp unde putem cauta userii dupa nume, ne trebuie o forma pentru a introduce numele, si acea forma sa trimita un request cu ce am scris, metoda ce va primi sa caute userii in baza de date si returneze aceeasi pagina, dar nu cu un atribut cu toata lista de useri, ci doar cu cei ceruti. Forma si metoda ar fi asa:

<**form:form** action="searchCustomer" method="get">  
 <label for="searchCustomer" style="display: inline;">Search customer</label>  
 <input id= "searchCustomer" type="text" name="customerName" placeholder="Name" class="searchBox"/>  
 <input type="submit" value="Search"/>  
</**form:form**>

@GetMapping("/searchCustomer")  
public String searchCustomer(@RequestParam("customerName")String customerName,Model model){  
 List<Customer> customers = customerService.searchCustomer(customerName);  
 model.addAttribute("customers",customers);  
  
 return "list-customers";  
}

Totusi, aici liber putem folosi si post, asa cum oricum trimitem param, si param il poate trimite si Post.

<**form:form** action="searchCustomer" method="post">  
 <label for="searchCustomer" style="display: inline;">Search customer</label>  
 <input id= "searchCustomer" type="text" name="customerName" placeholder="Name" class="searchBox"/>  
 <input type="submit" value="Search"/>  
</**form:form**>

@PostMapping("/searchCustomer")  
public String searchCustomer(@RequestParam("customerName")String customerName,Model model){  
 List<Customer> customers = customerService.searchCustomer(customerName);  
 model.addAttribute("customers",customers);  
  
 return "list-customers";  
}

Chiar daca forma are modelAttribute dar foloseste get, nu e nimic, pur si simplu nu va trimite acel atribut.

* Pentru forma, daca nu avem nevoie sa trimitem atribute, de multe ori putem folosi si post si get.
* **Atentie**! Butoanele separate de forma(<a>,<type=”submit” ...) mereu trimit get! Nu putem folosi PostMapping, caci ele mereu trimit doar get! Daca butonul face parte din forma, forma va alege metoda setata de noi, insa daca butonul e fara forma, mereu trimite get, asa cum nu-i putem seta metoda. De ex aici:
* <**c:forEach** var="customer" items="${customers}">  
    
   <**c:url** var="updateLink" value="/customer/showFormForUpdate">  
   <**c:param** name="customerId" value="${customer.id}"/>  
   </**c:url**>  
   <**c:url** var="deleteLink" value="/customer/delete">  
   <**c:param** name="customerId" value="${customer.id}"/>  
   </**c:url**>

butoanele astea nu fac parte din forma, deci mereu vor trimite GET. Nu putem folosi @PostMapping!

**Atentie**

Putem sa nu avem si o forma,dar numai prin forma putem specifica ce request vrem sa facem. Mereu cand o pagina face un request prin HTTP, adica sus in browser,sau cu un buton fara forma, se va cere GET automat. De ex, daca avem pur si simplu un buton fara forma care va trimite un request pentru a afisa o noua pagina ce are deja o forma, evident ca vom face un @GetMapping, asa cum el are nevoie sa fie trimise date, in cazul dat pagina si poate cu ceva date pentru forma.