

Oferecendo todos os temas

Transcrição

Por fim, vamos disponibilizar todos os temas para o usuário escolher o que ele quiser, não mais só o `aristo` ou `vader`. Para isso, no `TemaBean`, vamos criar o método `getTemas`, que retorna um array de Strings com todos os temas disponíveis:

```
public String[] getTemas() {  
    return new String[] { "afterdark", "afternoon", "afterwork", "aristo",  
        "black-tie", "blitzer", "bluesky", "bootstrap", "casablanca",  
        "cupertino", "cruze", "dark-hive", "delta", "dot-luv",  
        "eggplant", "excite-bike", "flick", "glass-x", "home",  
        "hot-sneaks", "humanity", "le-frog", "midnight", "mint-choc",  
        "overcast", "pepper-grinder", "redmond", "rocket", "sam",  
        "smoothness", "south-street", "start", "sunny", "swanky-purse",  
        "trontastic", "ui-darkness", "ui-lightness", "vader" };  
}
```

Mas como tornar isso dinâmico no template? Ao invés de criar um `f:selectItem` para cada tema, utilizaremos o componente `f:selectItems`, que irá receber todos os temas de uma vez só, através do método que acabamos de criar, para fazer um laço. Esse laço terá uma variável `tema`, que representa o tema a cada iteração, e essa variável será o valor dos atributos `itemLabel` e `itemValue`:

```
<p:themeSwitcher id="basic" style="width:165px" value="#{temaBean.tema}">  
    <f:selectItem itemLabel="Choose Theme" itemValue="" />  
    <f:selectItems value="#{temaBean.temas}" var="tema" itemLabel="#{tema}" itemValue="#{tema}" />  
    <f:ajax />  
</p:themeSwitcher>
```

Com isso conseguimos disponibilizar dinamicamente os itens para o usuário e finalizar este capítulo!