1 -Declarando managed bean

```
1.
2.  package bean;
3.
4.  import java.io.Serializable;
5.  import javax.faces.bean.ManagedBean;
6.  import javax.faces.bean.SessionScoped;
7.
8.  @ManagedBean
9.  @SessionScoped
10. public class User implements Serializable{
11.
12.     private String username;
13.     private String password;
14.
15.     public String getUsername() {
16.         return username;
17.     }
18.
19.     public void setUsername(String username) {
20.         this.username = username;
21.     }
22.
23.     public String getPassword() {
24.         return password;
25.     }
26.
27.     public void setPassword(String password) {
28.         this.password = password;
29.     }
30.
31.
32. }
```

- **Username e password são propriedades do bean;**
- **O escopo é do tipo sessionscoped. Escopos que podem ser associados ao bean:**
    1. Sessionscoped : o bean será mantido enquanto a sessão http esta ativa
    2. Requestscoped: o bean será mantido enquanto houver requisições http
    3. Viewscoped: o bean será mantido enquanto ele estiver na mesma view(pagina)
    4. Applicationscoped: o bean será mantido enquanto a aplicação esta online(o bean será acessível a todos)
    5. Nonescoped: o bean não será mantido nem inicializado no escopo porem ele pode ser acessado por outro bean herdando assim o escopo do bean que o requisitou.

2 - Inicializando managed bean

```
1.
2.  package bean;
3.
4.  import java.io.Serializable;
5.  import javax.annotation.PostConstruct;
6.  import javax.annotation.PreDestroy;
7.  import javax.faces.bean.ManagedBean;
8.  import javax.faces.bean.ManagedProperty;
9.  import javax.faces.bean.SessionScoped;
10.
11. @ManagedBean
12. @SessionScoped
13. public class User implements Serializable{
14.
15.     @ManagedProperty(value = "teste")
16.     private String username;
17.
18.     private String password;
19.
20.     @PostConstruct
21.     private void init(){
22.         System.out.println("iniciando o bean..usuario tem valor = "+username);
23.     }
24.
25.     @PreDestroy
26.     private void clean(){
27.         System.out.println("destruindo o bean.");
28.     }
29.
30.     public String getUsername() {
31.         return username;
32.     }
33.
34.     public void setUsername(String username) {
35.         this.username = username;
36.     }
37.
38.     public String getPassword() {
39.         return password;
40.     }
41.
42.     public void setPassword(String password) {
43.         this.password = password;
44.     }
45.
46.
47. }
```

- **@ManagedProperty inicializa uma propriedade do bean.**
- **@PostConstruct chama um método quando o bean é inicializado ou instanciado..**
- **@PreDestroy chama um método antes do bean ser destruído.**

3 - Gerenciando dependências de bean

JSF suporta inversão de controle(IoC) ou seja o managed bean pode ser acoplado em tempo de execução sem precisar manipular esse acoplamento no código da aplicação;

```
1.
2.  package bean;
3.
4.  import java.io.Serializable;
5.  import javax.faces.bean.ManagedBean;
6.  import javax.faces.bean.ManagedProperty;
7.  import javax.faces.bean.NoneScoped;
8.
9.  @ManagedBean(name = "profession")
10. @NoneScoped
11. public class Profession implements Serializable{
12.
13.     @ManagedProperty(value = "Software Engineer")
14.     private String title;
15.
16.     @ManagedProperty(value = "TI")
17.     private String industry;
18.
19.     public String getTitle() {
20.         return title;
21.     }
22.
23.     public void setTitle(String title) {
24.         this.title = title;
25.     }
26.
27.     public String getIndustry() {
28.         return industry;
29.     }
30.
31.     public void setIndustry(String industry) {
32.         this.industry = industry;
33.     }
34.
35.
36. }
37. /*#############################################################################################################*/
```

```
1.  package bean;
2.
3.  import java.io.Serializable;
4.  import javax.faces.bean.ManagedBean;
5.  import javax.faces.bean.ManagedProperty;
6.  import javax.faces.bean.SessionScoped;
7.
8.  @ManagedBean
9.  @SessionScoped
10. public class User implements Serializable{
11.
12.   @ManagedProperty("#{profession}")
13.   private Profession profession;
14.
15.     public Profession getProfession() {
16.         return profession;
17.     }
18.
19.     public void setProfession(Profession profession) {
20.         this.profession = profession;
21.     }
22.
23.
24.
25. }
```

Acessando managed beans através do código

```
1.
2.  package bean;
3.
4.  import java.io.Serializable;
5.  import javax.el.ELContext;
6.  import javax.el.ExpressionFactory;
7.  import javax.el.ValueExpression;
8.  import javax.faces.application.Application;
9.  import javax.faces.bean.ManagedBean;
10. import javax.faces.bean.SessionScoped;
11. import javax.faces.context.FacesContext;
12.
13. @ManagedBean
14. @SessionScoped
15. public class OutroBean implements Serializable{
16.
17.     public User getUser(){
18.
19.         FacesContext context = FacesContext.getCurrentInstance();
20.         Application application = context.getApplication();
21.         ELContext el = context.getELContext();
22.         ExpressionFactory ef = application.getExpressionFactory();
23.         ValueExpression value = ef.createValueExpression(el,"#{user}",User.class);
24.         User user = (User) value.getValue(el);
25.
26.         return user;
27.     }
28. }
```

Usando @Named e @Inject para pegar dados de outro bean

```
1.  package bean;
2.
3.  import java.io.Serializable;
4.  import javax.faces.bean.SessionScoped;
5.  import javax.inject.Inject;
6.  import javax.inject.Named;
7.
8.  @Named
9.  @SessionScoped
10. public class User implements Serializable{
11.
12.   @Inject
13.   private Profession profession;
14.
15.     public Profession getProfession() {
16.         return profession;
17.     }
18.
19.     public void setProfession(Profession profession) {
20.         this.profession = profession;
21.     }
22.
```

```
1.  }
```

```
2.
3.  package bean;
4.
5.  import java.io.Serializable;
6.  import javax.annotation.PostConstruct;
7.  import javax.inject.Named;
8.
9.  /**
10.  *
11.  * @author Deivid
12.  */
13. @Named
14. public class Profession implements Serializable {
15.
16.     private String title;
17.
18.     private String industry;
19.
20.     @PostConstruct
21.     public void ini() {
22.         this.title = "Software Engineer";
23.         this.industry = "TI";
24.     }
25.
26.     public String getTitle() {
27.         return title;
28.     }
29.
30.     public void setTitle(String title) {
31.         this.title = title;
32.     }
33.
34.     public String getIndustry() {
35.         return industry;
36.     }
37.
38.     public void setIndustry(String industry) {
39.         this.industry = industry;
40.     }
41.
42. }
```

```
23.
```