

Mãos à obra: Implementando a autorização

O primeiro passo na autenticação vai ser checar se o usuário já fez login, e se não for o caso ele deve ser redirecionado para a página de login. Para isso precisamos implementar um componente para interceptar as requisições, no mundo JSF usaremos o `PhaseListener` :

1) Crie um `PhaseListener` chamado `Autorizador` no pacote `br.com.caelum.livraria.util` :

```
public class Autorizador implements PhaseListener {

    private static final long serialVersionUID = 1L;

    @Override
    public void afterPhase(PhaseEvent event) {
    }

    @Override
    public void beforePhase(PhaseEvent event) {
    }

    @Override
    public PhaseId getPhaseId() {
        return PhaseId.RESTORE_VIEW;
    }
}
```

2) No método `afterPhase(event)` precisamos ter em mãos o nome do arquivo `.xhtml` que estamos acessando para podermos verificar se é a página de login. Se for, não precisamos fazer nada nesse momento:

```
@Override
public void afterPhase(PhaseEvent event) {
    FacesContext context = event.getFacesContext();
    String nomePagina = context.getViewRoot().getViewId();

    if("/login.xhtml".equals(nomePagina)) {
        return;
    }
}
```

3) Precisamos também verificar se o usuário fez login, ou seja, se há um usuário na sessão não precisamos fazer nada:

```
Usuario usuarioLogado = (Usuario) context.getExternalContext()
    .getSessionMap().get("usuarioLogado");

if(usuarioLogado != null) {
    return;
}
```

4) Porém, se não houver precisamos redirecioná-lo para a página de login. Para isso, vamos usar o `NavigationHandler` :

```

NavigationHandler handler = context.getApplication().getNavigationHandler();
handler.handleNavigation(context, null, "/login?faces-redirect=true");

context.renderResponse();

```

5) O método completo deverá ficar assim:

```

@Override
public void afterPhase(PhaseEvent event) {

    FacesContext context = event.getFacesContext();
    String nomePagina = context.getViewRoot().getViewId();

    System.out.println(nomePagina);

    if ("/login.xhtml".equals(nomePagina)) {
        return;
    }

    Usuario usuarioLogado = (Usuario) context.getExternalContext()
        .getSessionMap().get("usuarioLogado");

    if(usuarioLogado != null) {
        return;
    }

    NavigationHandler handler = context.getApplication().getNavigationHandler();
    handler.handleNavigation(context, null, "/login?faces-redirect=true");

    context.renderResponse();
}

```

6) Repare que estamos buscando o *usuarioLogado* no *SessionMap*. Em algum momento é preciso disponibilizar o *usuarioLogado* no mapa, esse lugar é o método *efetuaLogin* do *LoginBean*. Altere o método para ficar assim:

```

public String efetuaLogin() {
    System.out.println("Fazendo login do usuário "
        + this.usuario.getEmail());

    FacesContext context = FacesContext.getCurrentInstance();
    boolean existe = new UsuarioDao().existe(this.usuario);

    if (existe) {

        context.getExternalContext().getSessionMap()
            .put("usuarioLogado", this.usuario);

        return "livro?faces-redirect=true";
    }

    //tbm pode devolver o nome da página
    //return "login";
    return null;
}

```

7) O JSF precisa conhecer nosso Autorizador . Vamos registrá-lo no faces-config.xml :

```
<phase-listener>br.com.caelum.livraria.util.Autorizador</phase-listener>
```

Caso tudo esteja correto, ao tentar acessar livro.xhtml ou autor.xhtml sem estarmos autenticados seremos redirecionados à página de login.