✅ Como criar e registrar um PhaseListener no JSF

📌 1. Criar a classe que implementa `PhaseListener`

Crie uma nova classe Java que implementa a interface `javax.faces.event.PhaseListener`. Você precisa implementar três métodos:

📄 Exemplo completo da classe `TestPhaseListener`:

```
package br.jsf2.listener;

import javax.faces.event.PhaseEvent;
import javax.faces.event.PhaseId;
import javax.faces.event.PhaseListener;

public class TestPhaseListener implements PhaseListener {

    @Override
    public void afterPhase(PhaseEvent event) {
        event.getFacesContext().getExternalContext()
            .log("AFTER: " + event.getPhaseId());
    }

    @Override
    public void beforePhase(PhaseEvent event) {
        event.getFacesContext().getExternalContext()
            .log("BEFORE: " + event.getPhaseId());
    }

    @Override
    public PhaseId getPhaseId() {
        return PhaseId.ANY_PHASE; // Escuta todas as fases do ciclo de vida
    }
}
```

🧠 Explicação dos métodos:

Método	Descrição
`beforePhase()`	Executa **antes** de cada fase do ciclo de vida JSF.
`afterPhase()`	Executa **depois** de cada fase.

`getPhaseId()`	Indica **quais fases** o listener irá escutar. Ao retornar `PhaseId.ANY_PHASE`, o listener será chamado para todas as fases.

📌 2. Registrar o `PhaseListener` no `faces-config.xml`

No arquivo `faces-config.xml`, registre a classe que você acabou de criar dentro da tag `<lifecycle>`:

```
<lifecycle>
   <phase-listener>
      br.jsf2.listener.TestPhaseListener
   </phase-listener>
</lifecycle>
```

> Coloque isso **fora** das tags `<application>` ou `<managed-bean>`, no nível raiz do `faces-config.xml`.

🎯 Resultado Esperado

Ao rodar a aplicação JSF com esse `PhaseListener`, você verá no **console** ou nos **logs do servidor** mensagens como:

```
BEFORE: RESTORE_VIEW 1
AFTER: RESTORE_VIEW 1
BEFORE: APPLY_REQUEST_VALUES 2
AFTER: APPLY_REQUEST_VALUES 2
...
```

🧪 Dica extra para testes

Para visualizar esse comportamento na prática, crie uma página simples com um formulário e botão:

```
<h:form>
   <h:commandButton value="Testar Ciclo JSF" action="#{aluno.salvar}" />
</h:form>
```

E no `AlunoBean`:

```
public String salvar() {
```

```
    System.out.println("Método salvar() executado.");
    return null;
}
```