

Le Design Pattern MVC



Philippe Mathieu & Guillaume Dufrene

IUT-A Lille

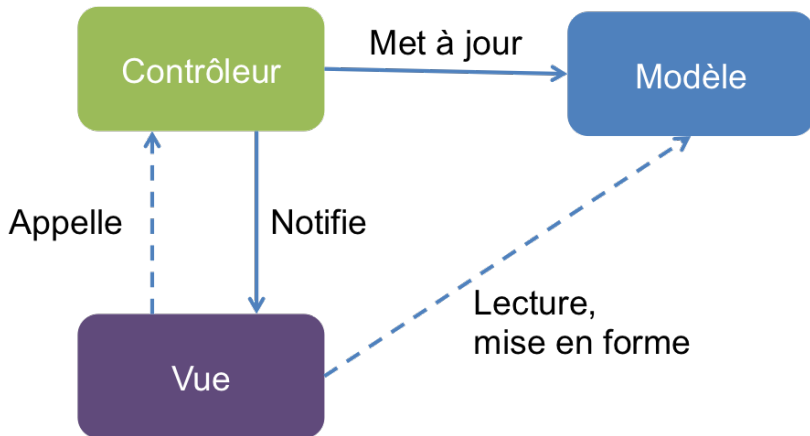
<http://www.iut-a.univ-lille.fr>

prenom.nom@univ-lille.fr

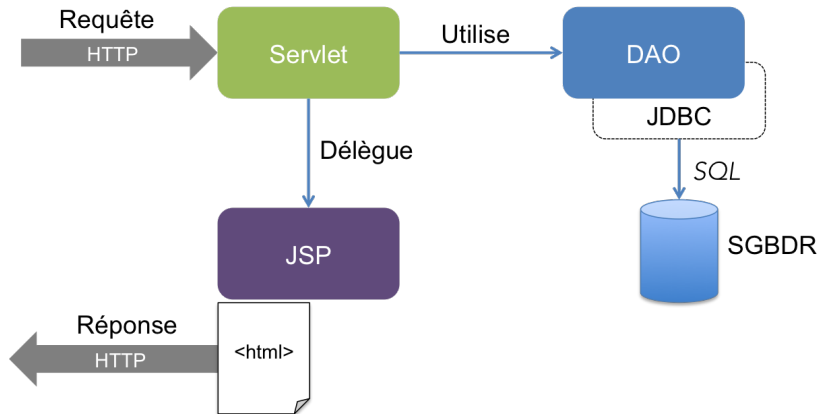
Modèle - Vue - Contrôleur (MVC)

- Séparer les préoccupations
 - ▶ la logique de traitement,
 - ▶ l'accès au données,
 - ▶ l'affichage
- Faciliter le travail des développeurs :
 - ▶ Découpage et répartition du travail,
 - ▶ Organisation du code pour la maintenance,
 - ▶ Indépendance des objets pour l'évolution de l'application

Pattern général MVC



Application du MVC au web JEE

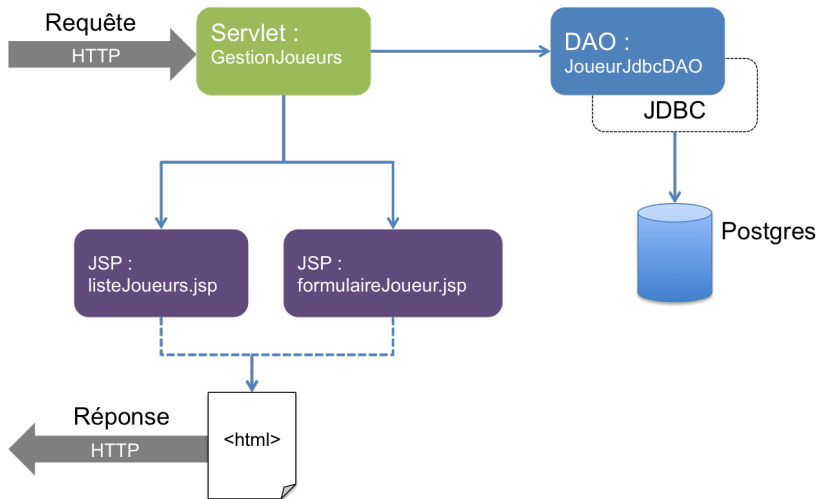


Un exemple de mise en oeuvre

- Ajouter un joueur
- Modifier un joueur existant
- Supprimer un joueur
- Afficher la liste des joueurs par club

Joueur
id
nom
prenom
poste
club
dateNaiss

Implémentation MVC correspondante



```
class JoueurJdbcDao
{
    JoueurJdbcDao(DS ds) { ... }
    public void create( Joueur joueur ) { ... }
    public void update( Joueur joueur ) { ... }
    public void delete( int idJoueur ) { ... }
    public Joueur find( int idJoueur ) { ... }
    public List<Joueur> findByClub( String club ) { ... }
}
```

```
@WebServlet("/GestionJoueurs")
public class GestionJoueurs extends HttpServlet
{
    JoueurJdbcDao daoJoueur;

    public void service(...) {
        String action = req.getParameter("act");
        String vue = "WEB-INF/formulaireJoueur.jsp";
        switch( action ) {
            case "lister":
                vue = "WEB-INF/listeJoueurs.jsp";
                listerParClub(req); break;
            case "editer": editer(req); break;
            case "modifier":
                if ( modifier(req) )
                { res.sendRedirect("joueurs?act=lister"); return; }
                break;
            default:
                res.sendError( "Action non supportée" ); return;
        }
        res.getRequestDispatcher( vue ).forward( req, res ); }
    // ...
}
```



```
private void listerParClub( HttpServletRequest req )
{ String club = req.getParameter("club");
  if ( club == null ) club = "Lille";
  List<Joueur> joueurs = daoJoueur.findByClub(club);
  req.setAttribute( "joueurs", joueurs );
}

private boolean modifier( HttpServletRequest req )
{ try
  {
    Joueur j = joueurFromRequest( req );
    if ( j.id == null ) daoJoueur.create(club);
    else daoJoueur.update(club);
  }
  catch( Exception e )
  { req.setAttribute( "error", e );
    req.setAttribute( "joueur", j );
    return false;
  }
  return true;
}
```

```
<% Joueur j = (Joueur) request.getAttribute("joueur"); %>
<h1><%= (j == null ? "Ajouter" : "Modifier") %> un joueur.</h1>
<form action=GestionJoueurs method=POST>
  <input type='hidden' name='act' value='modifier'>
  <% if ( j != null ) { %>
    Id : <%= j.id %>
    <input type='hidden' name='id' value='<%= j.id %>' /> <br/>
  <% } %>
  Nom      : <input type='text' name='nom'      value='<%= j.nom %>' />
  Prenom   : <input type='text' name='prenom'   value='<%= j.prenom %>' />
  // ...
  <input type='submit' value='Enregistrer' />
</form>
```

- MVC est un patron de conception couramment utilisé pour faciliter le développement d'applications.
- Sa mise en oeuvre web peut être implémenté en JEE par :
 - ▶ Une servlet qui orchestre un ensemble de traitements
 - ▶ Un ou plusieurs DAO pour manipuler le modèle, généralement via JDBC.
 - ▶ Des JSP qui mettent en forme le modèle et permettent à l'utilisateur d'effectuer des actions.