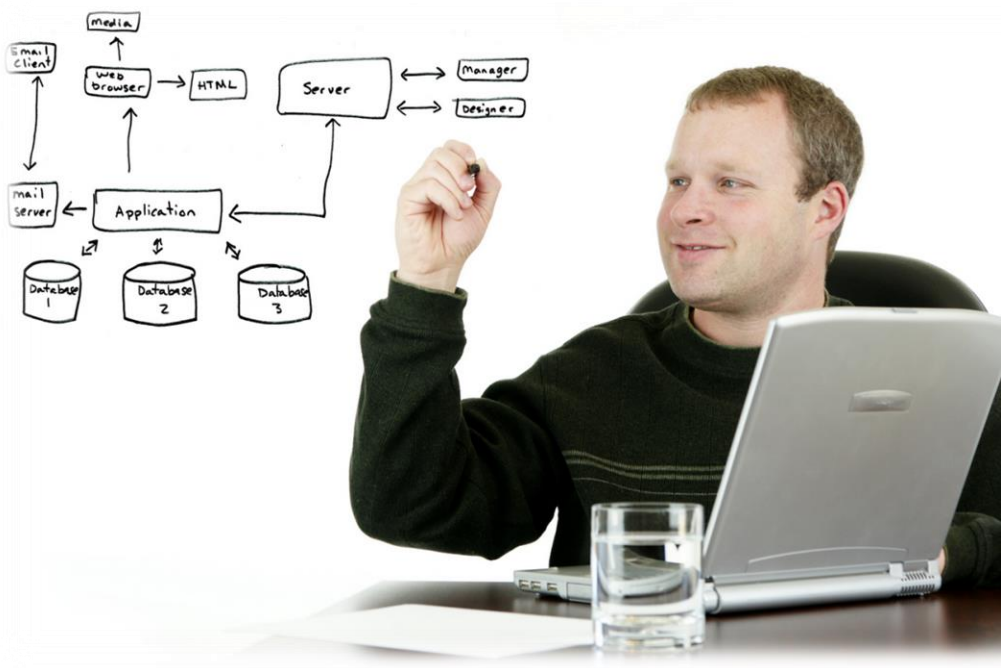


Présentation - Architecture applicative

Max Devulder

Découpage projet

Découpage logique

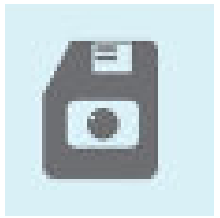
Présentation

- Affichage.
- Interaction avec utilisateur
- Capture / validation des données

Métier

- Règles de gestions
- Logique applicative

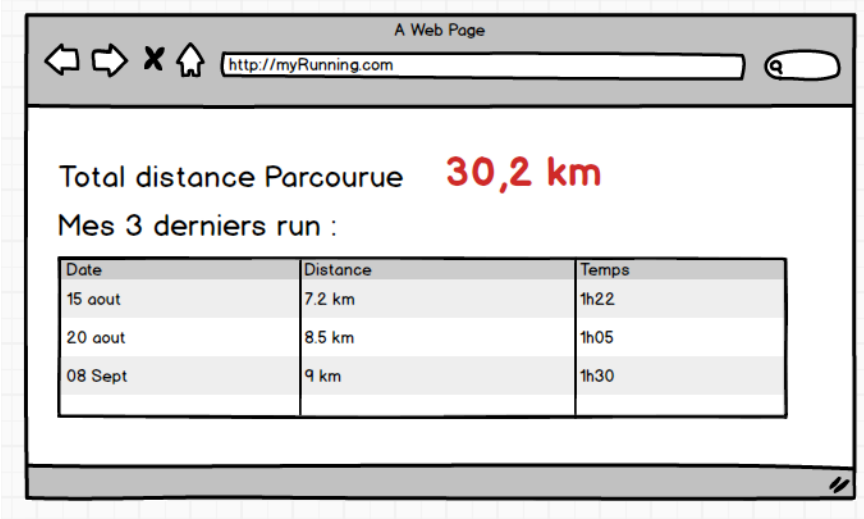
Persistance



- Accès base de données

Découpage projet

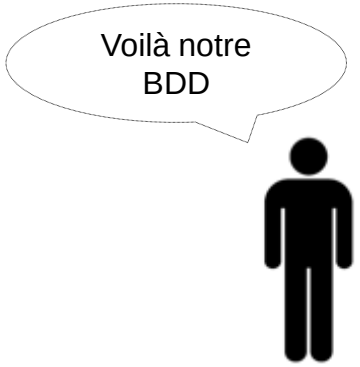
Un cas concret



TBL_SESSION					
ID	DISTANCE	Année	Mois	Jours	TEMPS
1	5502,6	2018	05	02	60
2	7203,5	2018	08	15	82
3	8501,9	2018	08	20	65
4	9002,5	2018	09	08	90

MOA

Vous



D.B.A

Partie Front : HTML / JS / CSS

home.html

```
<div>
   <span> Total distance parcourue </span><span id=distance></span>
</div>

<table>
 <thead></thead>
 <tbody>….</tbody>
<table>
```

Présentation | Métier | Persistance

Un Controller et un (ou plusieurs) beans

Bean représentant ma page web DTO (Data Transfert Object)

```java
public class RunningDO {

  private Long id;
  private Float distance;
  private Integer annee;
  private Integer mois;
  private Integer jour;
  private Float temps;

  // + Accesseurs (GET/SET)

}
```

```java
public class HomeDTO {

  private Float distanceTotale;

  private List<RunningDTO> derniersRun;

  // + Accesseurs (GET/SET)
}
```

```java
public class RunningDTO {

  private String date;
  private Float distanceKm;
  private String temps;

  // + Accesseurs (GET/SET)

}
```

Présentation | **Métier** | Persistance

Découpage projet

Un cas concret

Un Controller et un (ou plusieurs) beans

- Validation de surface (optionel)
- Si OK, elle appelle du ou des différents services (BO)

```java
@WebServlet("/HomeServlet")
public class HomeServlet extends HttpServlet {

  private RunningBO runningBO;

  public HomeServlet() {
    runningBO = new RunningBO();
  }
```

```java
@Override
protected void doGet(final HttpServletRequest request, final HttpServletResponse response)
    throws ServletException, IOException {

  // Appel des différentes données
  final Float distanceTotale = runningBO.countDistance();
  final List<RunningDO> derniersRun = runningBO.findLast(3);

  // Construction du DTO
  final HomeDTO dto = new HomeDTO();
  dto.setDistanceTotale(distanceTotale);
  for (final RunningDO runningDO : derniersRun) {
    final RunningDTO runningDTO = new RunningDTO();
    runningDTO.setDistanceKm(runningDO.getDistance());
    runningDTO.setTemps(runningDO.getTemps() / 60 + "h" + runningDO.getTemps() % 60);
    runningDTO.setDate(runningDO.getJour() + " " + CalendarUtils.getMonth(runningDO.getMois()));
    dto.getDerniersRun().add(runningDTO);
  }

  // Retours
  request.getSession().setAttribute("home", dto);
}
```

Présentation | **Métier** | Persistance

Découpage projet

Un cas concret

Le service (BO) prépare un nouvel utilisateur (DO) à persister et demande à la couche Persistance (DAO) de s'en occuper.

```java
public class RunningService {

  private RunningDAO dao;

  public RunningService() {
    dao = new RunningDAO();
  }
```

```java
  public List<RunningDTO> findLast(final Integer number) {
    // Recherche information en base
    final List<RunningDO> entities = dao.findLast(number);

    // Mapping
    final List<RunningDTO> listeRunning = new ArrayList<>(number);
    for (final RunningDO runningDO : entities) {
      final RunningDTO runningDTO = map(runningDO);
      listeRunning.add(runningDTO);
    }
    return listeRunning;

  }
```

```java
private RunningDTO map(final RunningDO runningDO) {
  final RunningDTO runningDTO = new RunningDTO();
  runningDTO.setDistanceKm(runningDO.getDistance());
  runningDTO.setTemps(runningDO.getTemps() / 60 + "h" + runningDO.getTemps() % 60);
  runningDTO.setDate(runningDO.getJour() + " " + CalendarUtils.getMonth(runningDO.getMois()));
  return runningDTO;
}
```

Correspondance Formulaire/BDD = Logique métier

Présentation | **Métier** | Persistance

La couche Persistance (DAO) sauvegarde l'objet (DO) en base (Hibernate, JDBC….)

Seule couche à accéder à la BDD
Seule couche à connaitre le modèle physique

```
public Float countDistance() {
  final String sqlStatement = "SELECT SUM(DISTANCE) AS TOTAL FROM TBL_SESSION";
  PreparedStatement smt;
  try {
    smt = datasource.getConnection().prepareStatement(sqlStatement);
    final ResultSet rs = smt.executeQuery();
    while (rs.next()) {
      return rs.getFloat("TOTAL");
    }
  } catch (final SQLException e) {
    // Error managment here
  }
  return null;
}
```

Présentation | Métier | Persistance

La couche Persistance (DAO) sauvegarde l'objet (DO) en base (Hibernate, JDBC….)

Seule couche à accéder à la BDD
Seule couche à connaitre le modèle physique

```
public List<RunningDO> findLast(final Integer number) {
  final String sqlStatement = "SELECT * FROM TBL_SESSION ORDER BY ID DESC LIMIT " + number;
  final List<RunningDO> result = new ArrayList<>(number);
  PreparedStatement smt;
  try {
    smt = datasource.getConnection().prepareStatement(sqlStatement);
    final ResultSet rs = smt.executeQuery();
    while (rs.next()) {
      final RunningDO runningDO = new RunningDO();
      runningDO.setId(rs.getLong("ID"));
      runningDO.setDistance(rs.getFloat("DISTANCE"));
      runningDO.setAnnee(rs.getInt("TOTAL"));
      runningDO.setAnnee(rs.getInt("MOIS"));
      runningDO.setAnnee(rs.getInt("JOUR"));
      runningDO.setTemps(rs.getFloat("TEMPS"));

      result.add(runningDO);
    }
  } catch (final SQLException e) {
    // Error managment here
  }
  return result;
}
```

Présentation | Métier | Persistance

Découpage projet

Un cas concret

Couche applicative :

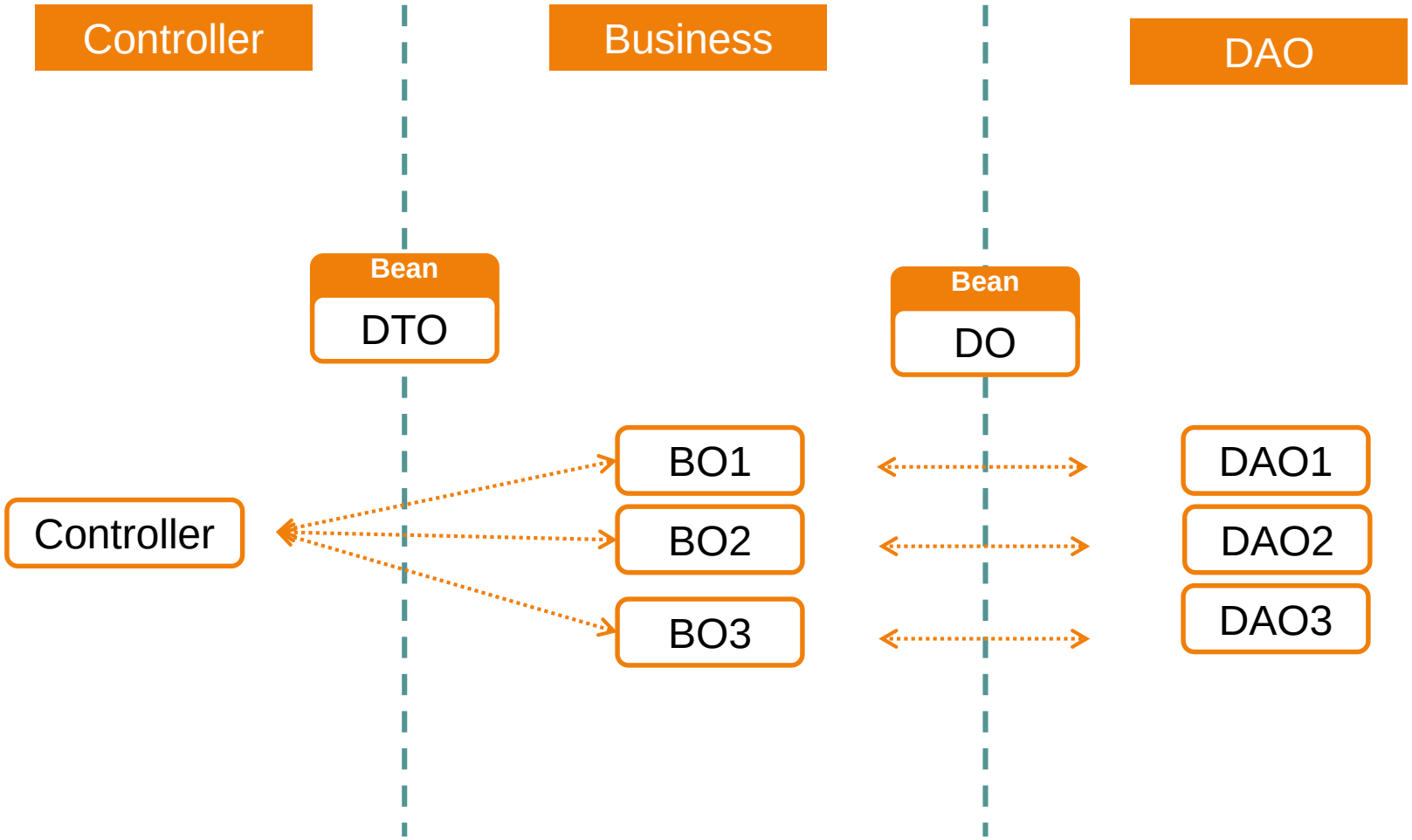
- 1 couche = 1 usage !
- Objets contenant les **traitements** de notre application, singleton.
- Couche Web : Validation de surface
- Couche Business : Code métier
- Couche Persistance : Gestion BDD.

Bean POJO (Plain Old Java Object) :

- Formule magique = Attributs + getter + setter et rien d'autre !
- Objets contenant les **données** de notre application, singleton.
- Servent de communication entre 2 couches.
- Permet l'isolement applicatif

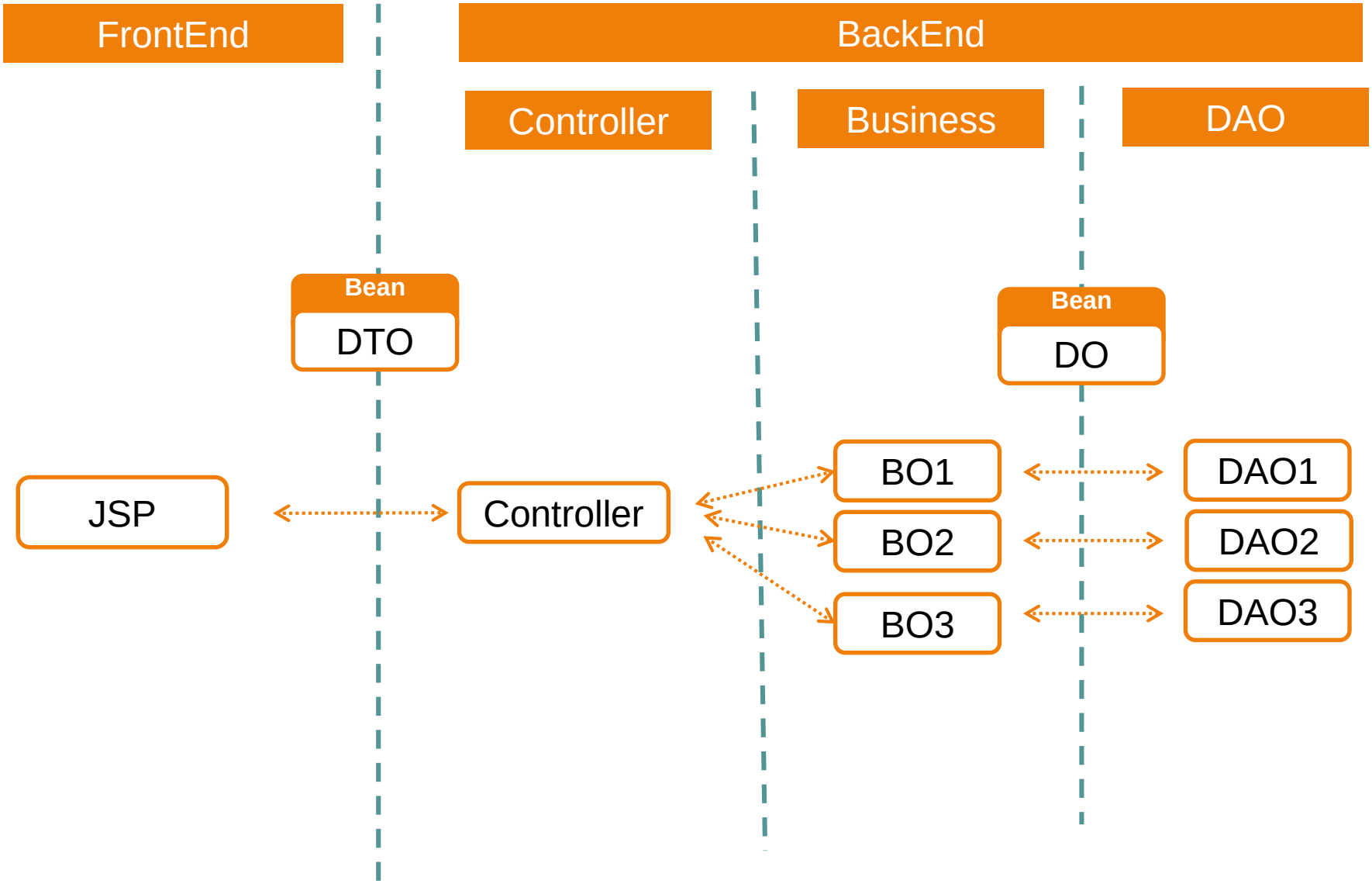
Découpage projet

Découpage technique



Découpage projet

Découpage technique



Présentation Frameworks et Outils

Le trio fondamental

Struts

- Implémentation du pattern MVC
- Définition de la configuratic… …hier XML

DEPRECATED

Hibernate

- Implémentation objet d'une BDD relationnelle.
- Dissociation entre SGBD physique et schéma object logique
- Remplace les accès BDD par des appels à des méthodes objet de haut niveau.
- Définition du mapping beans/table par fichier XML ou annotation

Spring

- Fabrique générique et unique d'objets.
- Définition de beans par fichier XML ou annotation