Université de Bouira – Département Informatique

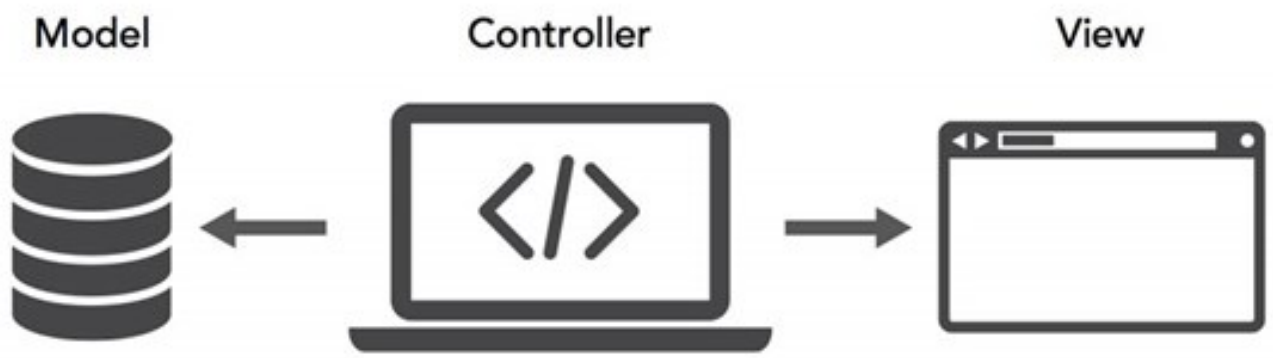
Matser Génie des systèmes Informatiques
Méthodes et Technologies de l'implémentation

# Approche MVC
# Model-View-Controller

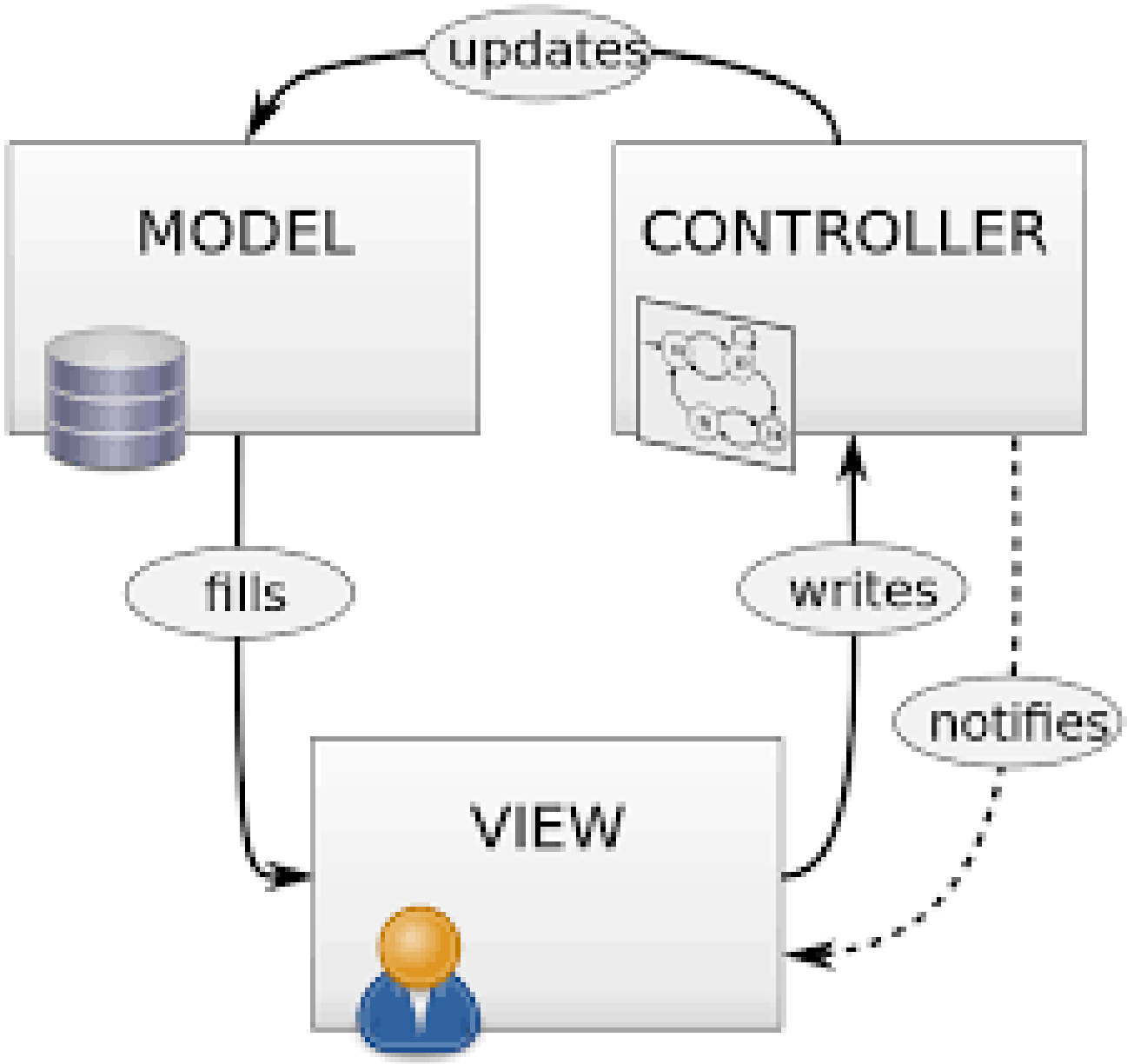
Taha Zerrouki
Taha.zerrouki @ gmail.com

# Définition

- **L'architecture MVC (modèle, vue et contrôleur) est un concept très puissant qui intervient dans la réalisation d'une application.**
- **Son principal intérêt est la séparation :**
  - des données (modèle),
  - de l'affichage (vue)
  - et des actions (contrôleur).



# MVC



# Exemple

- On veut créer un carnet des contacts
- Avec une fonctionnalité de recherche
-

# Code en Python

```
# Cherche un num dans un tableau par son nom
annuaire = [
    {'prenom':'Ahmed',  'nom':'Mehdi', 'tel':'0778787887'},
    {'prenom':'Mohamed', 'nom':'Rabehi','tel':'0778787887'},
    ]
def main():
    # lire des données a partir du clavier
    print "Recherche d'un telephone"
    print "Introduire Un nom"
    nom = raw_input()
    # nombre d'elements trouvés
    nb_found = 0
    # parcours des noms
    for personne in annuaire:
         # afficher toutes les personnes qui ont le nom donné
        if personne['nom'] == nom:
            print nom, personne['prenom'], personne['tel']
            nb_found += 1
    if not nb_found:
        print "ce nom %s n'existe pas "%nom
```

# Code en Python

```
# Cherche un num dans un tableau par son nom
annuaire = [
    {'prenom':'Ahmed',  'nom':'Mehdi', 'tel':'0778787887'},
    {'prenom':'Mohamed', 'nom':'Rabehi','tel':'0778787887'},
    ]
def main():
    # lire des données a partir du clavier
    print "Recherche d'un telephone"
    print "Introduire Un nom"
    nom = raw_input()
    # nombre d'elements trouvés
    nb_found = 0
    # parcours des noms
    for personne in annuaire:
        # afficher toutes les personnes qui ont le nom donné
        if personne['nom'] == nom:
            print nom, personne['prenom'], personne['tel']
            nb_found += 1
    if not nb_found:
        print "ce nom %s n'existe pas "%nom
```

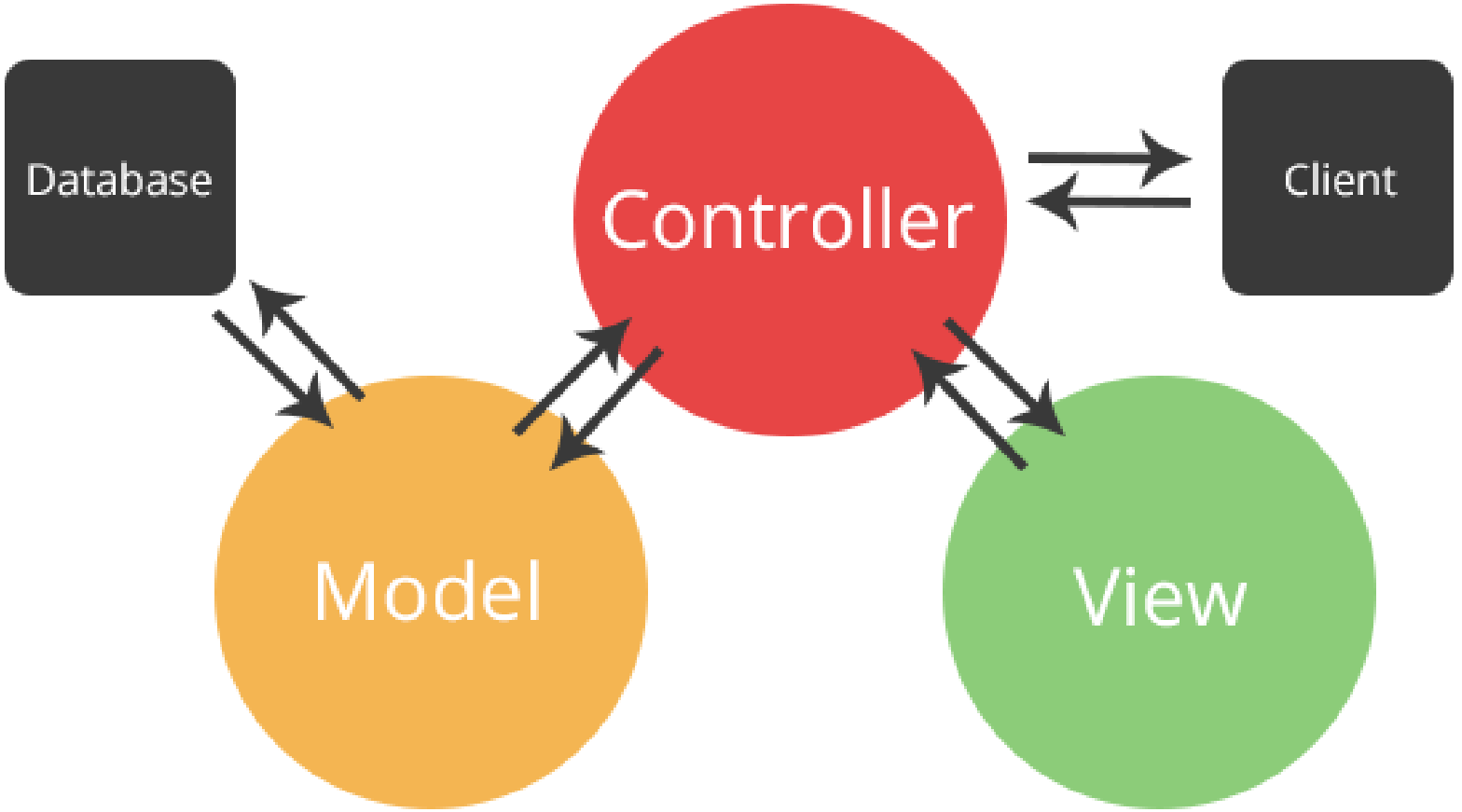
Données

Interface

Traitement

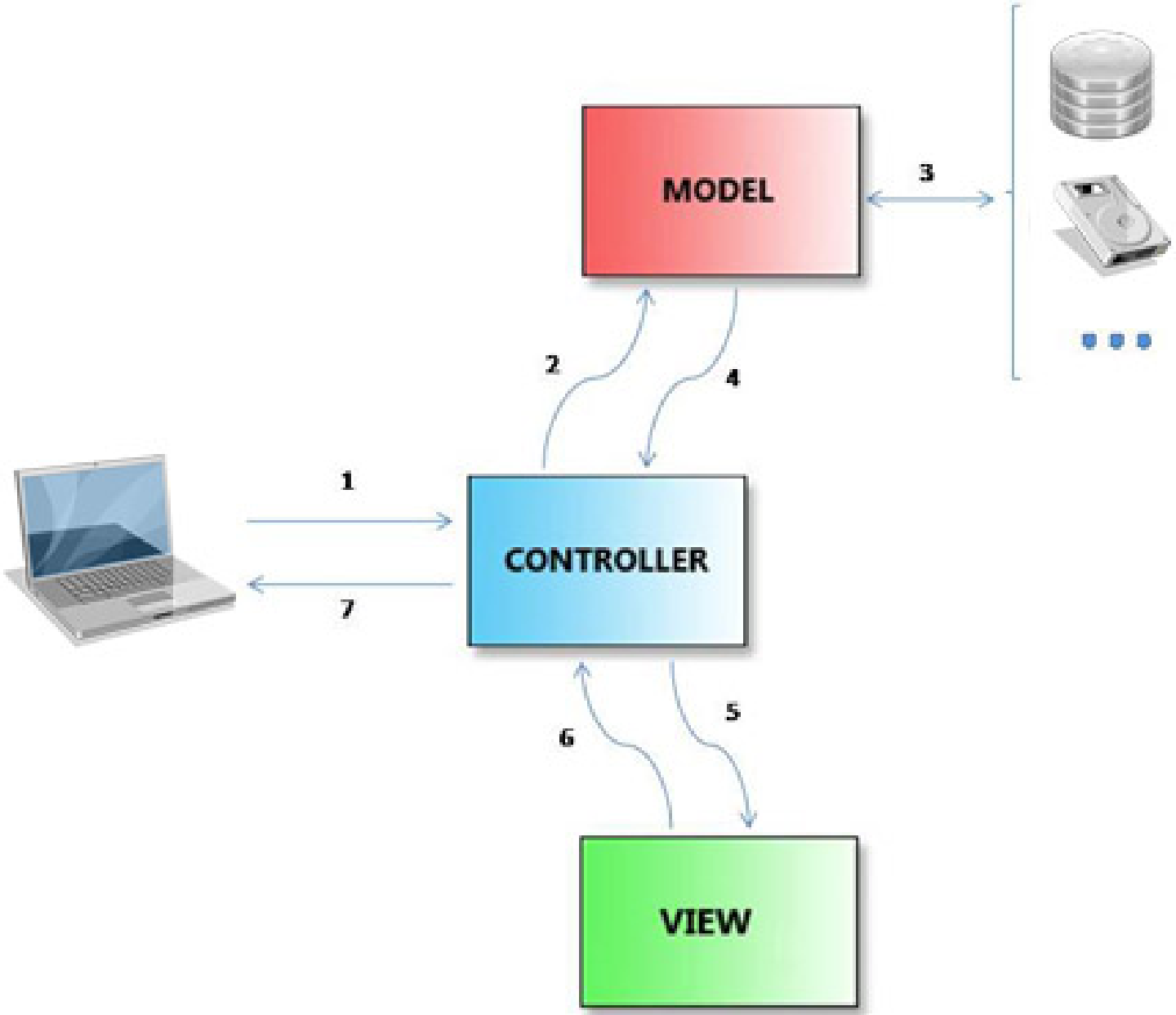
Interface

# MVC




AtomsNetwork

# MVC



# •Avantages du MVC

- **L'approche MVC apporte de réels avantages:**
  - **Une conception claire** et efficace grâce à la séparation des données de la vue et du contrôleur
  - **Un gain de temps de maintenance** et d'évolution du site
  - **Une plus grande souplesse** pour organiser le développement entre différents développeurs (indépendance des données, de l'affichage (webdesign) et des actions)

# •Inconvénients

- **L'inconvénient majeur du modèle MVC n'est visible que dans la réalisation de petits projets, de sites internet de faible envergure.**
- **En effet, la séparation des différentes couches nécessite la création de plus de fichiers:**
  - Un fichier pour le modèle
  - Un fichier pour le contrôleur
  - Un fichier pour la vue
- **Il n'est donc pas très intéressant de recourir à ce système dans ce cas**

# •MVC, comment ça marche?

- L’architecture MVC est donc décomposée en trois étapes:
- 1. Le Modèle
- 2. Le contrôleur
- 3. La vue

# •1. Le Modèle

- Le modèle correspond aux données,
- la plupart du temps stockées dans une base de données.
- Mais celles-ci peuvent également êtres contenues dans un fichier XML ou dans des fichiers texte.
- Les données peuvent être exploitées sous forme de classes, dans un langage de programmation orientée objet (Java, Python, PHP5).

# Exemple Modèle

```
class model:
    """ classe de modele de donnée"""
    def __init__(self):
        self.annuaire = [
        {'prenom':'Ahmed',  'nom':'Mahdi',
'tel':'0778787887'},
        {'prenom':'Mohamed',
'nom':'Mahdi','tel':'0778787887'},
        ]
    def rechercher(self, nom):
        """ rechecher un tel par nom"""
        …
        # La liste des personnes trouvées
        return personnes
```

Données
un table

# Exemple Modèle

```
class model_fichier:
    """ classe de modele de donnée"""
    def __init__(self):

        self.annuaire = []
        try:
            myfile = open("annuaire.txt")
        except:
            print "Can't open DataFile"
            sys.exit()
        lines = myfile.readlines()
      ...
    def rechercher(self, nom):
        """ rechecher un tel par nom"""
       ....
        # La liste des personnes trouvées
        return personnes
```

Données
Fichie

# •3. La vue

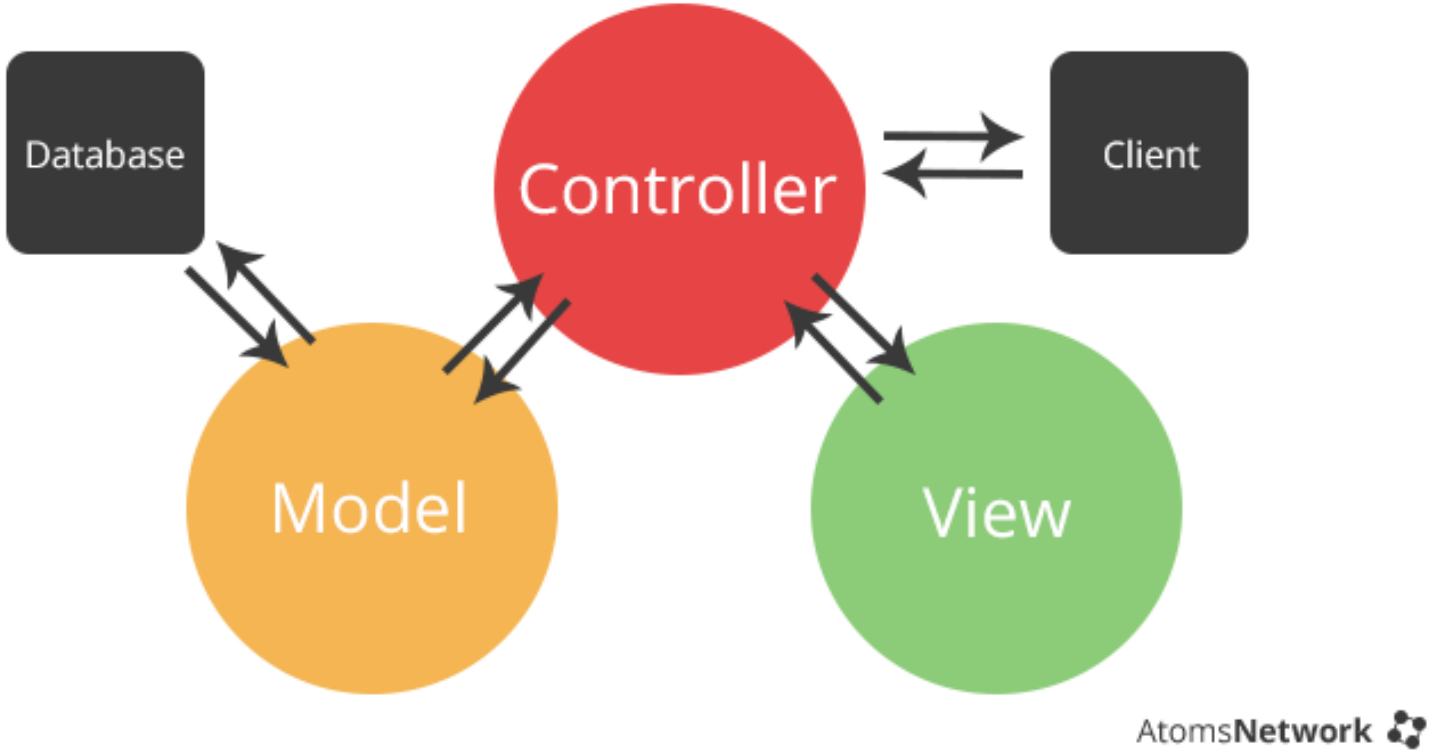
- la Vue correspond à l'interface avec laquelle l'utilisateur interagit.
- Les résultats renvoyés par le modèle sont dénués de toute présentation mais sont présentés par les vues.
- Plusieurs vues peuvent afficher les informations d'un même modèle.
- Elle peut être conçue en html, ou tout autre " langage " de présentation.
- La vue n'effectue aucun traitement, elle se contente d'afficher les résultats des traitements effectués par le modèle, et de permettre à l'utilisateur d'interagir avec elles.
-

# Example

```
class view:
    def __init__(self,):
        pass
    def input(self,):
        """ recuperer le nom à rechercher"""
        print "Recherche d'un telephone"
        print "Introduire Un nom"
        nom = raw_input()
        return nom

    def output(self, personnes):
        """ afficher les informations d'une liste des
personnes"""
        print "La liste des noms trouvés"
        print " %d personnes trouvées"%len(personnes)
        for pers in personnes:
            print pers['nom'], pers['prenom'], pers['tel']
```

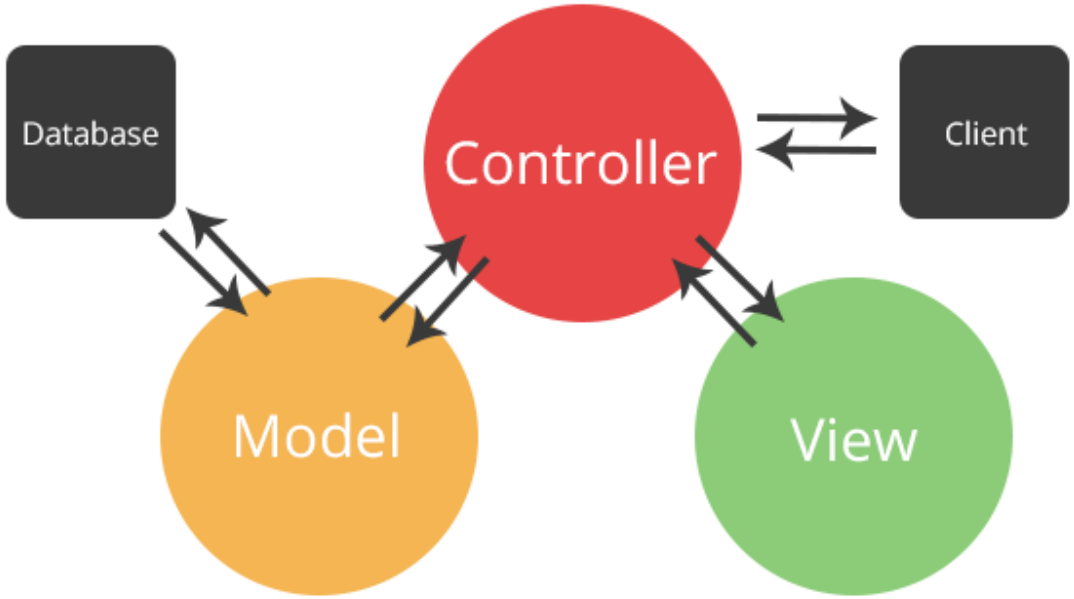
# •Le Contrôleur

- Le contrôleur est l'élément qui va utiliser les données pour les envoyer à la vue.
-
-
-



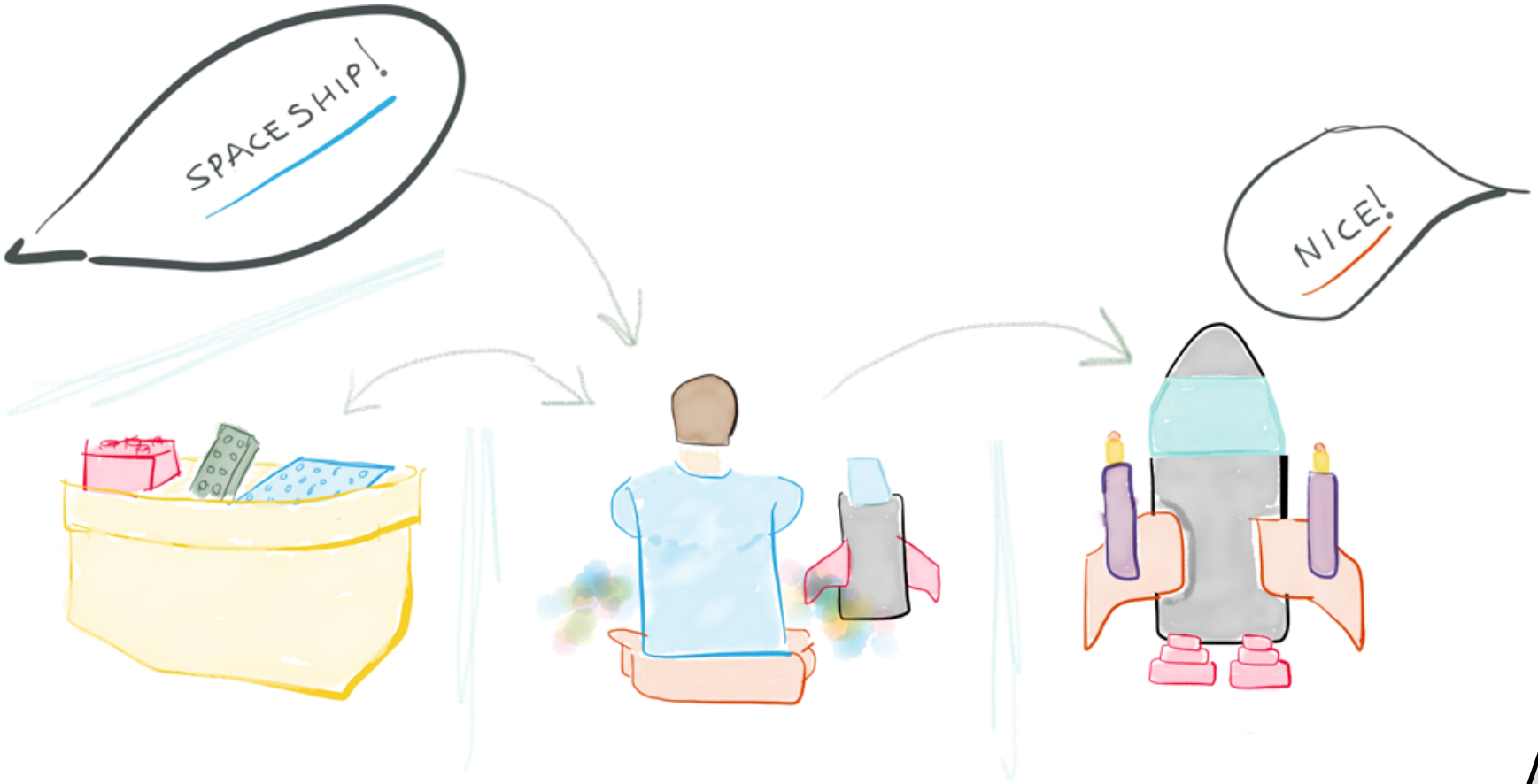
# •Le Contrôleur

- **Son rôle est donc de**
  - récupérer les informations,
  - de les traiter en fonction des paramètres demandés par la vue (par l'utilisateur, exemple: afficher les derniers articles),
  - puis de renvoyer à la vue les données afin d'être affichées.



# •Le Contrôleur

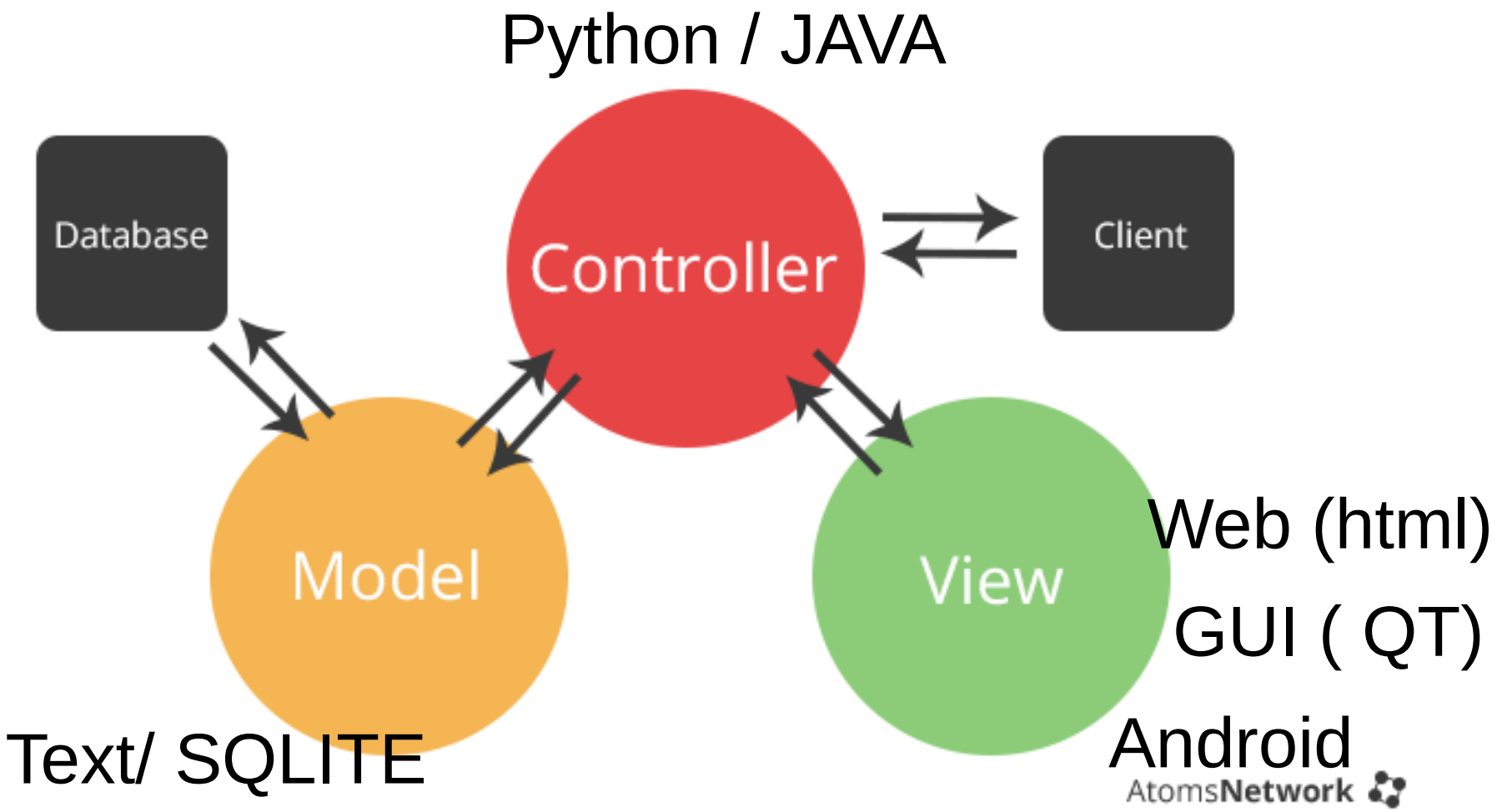
- **Le contrôleur peut donc :**
  - instancier différents objets (classe User, classe Articles, …)
  - qui enverront des requêtes vers la base de données ou récupéreront des données XML.



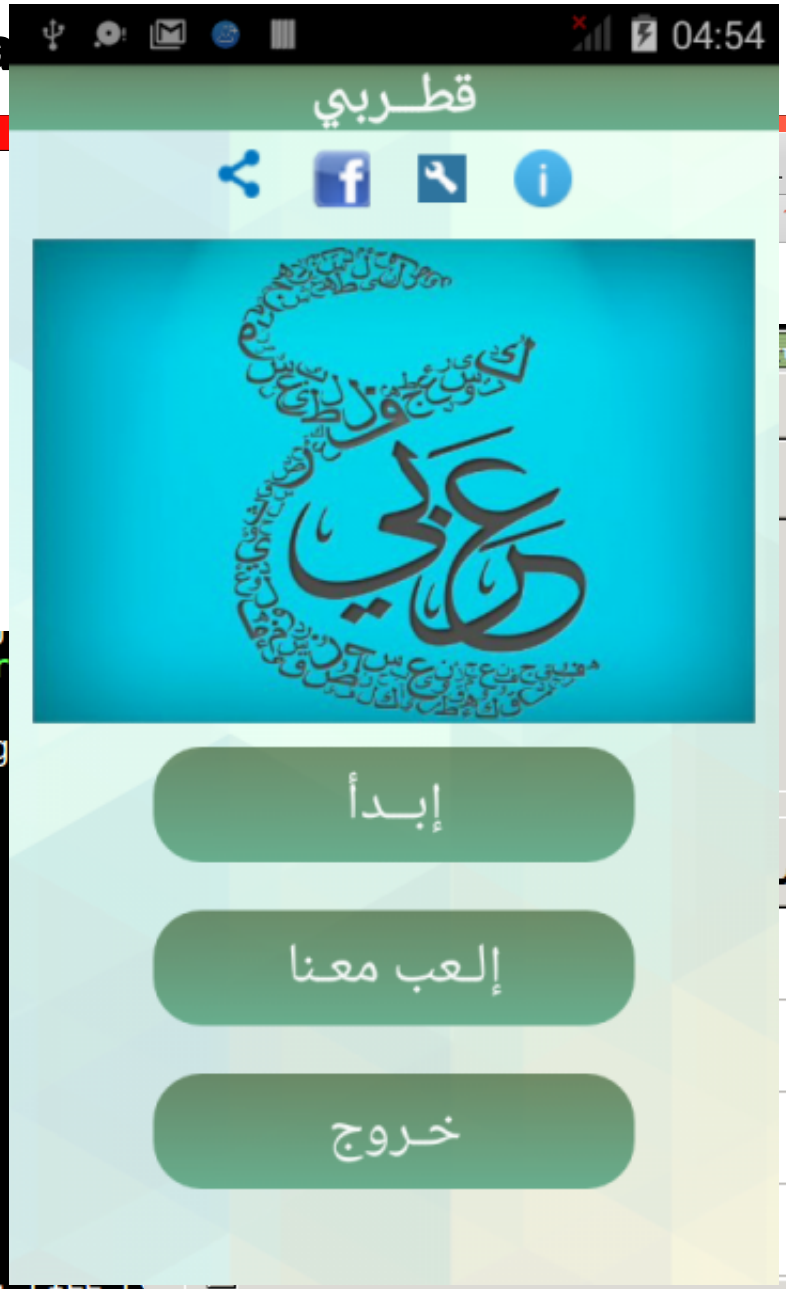
# Exemple

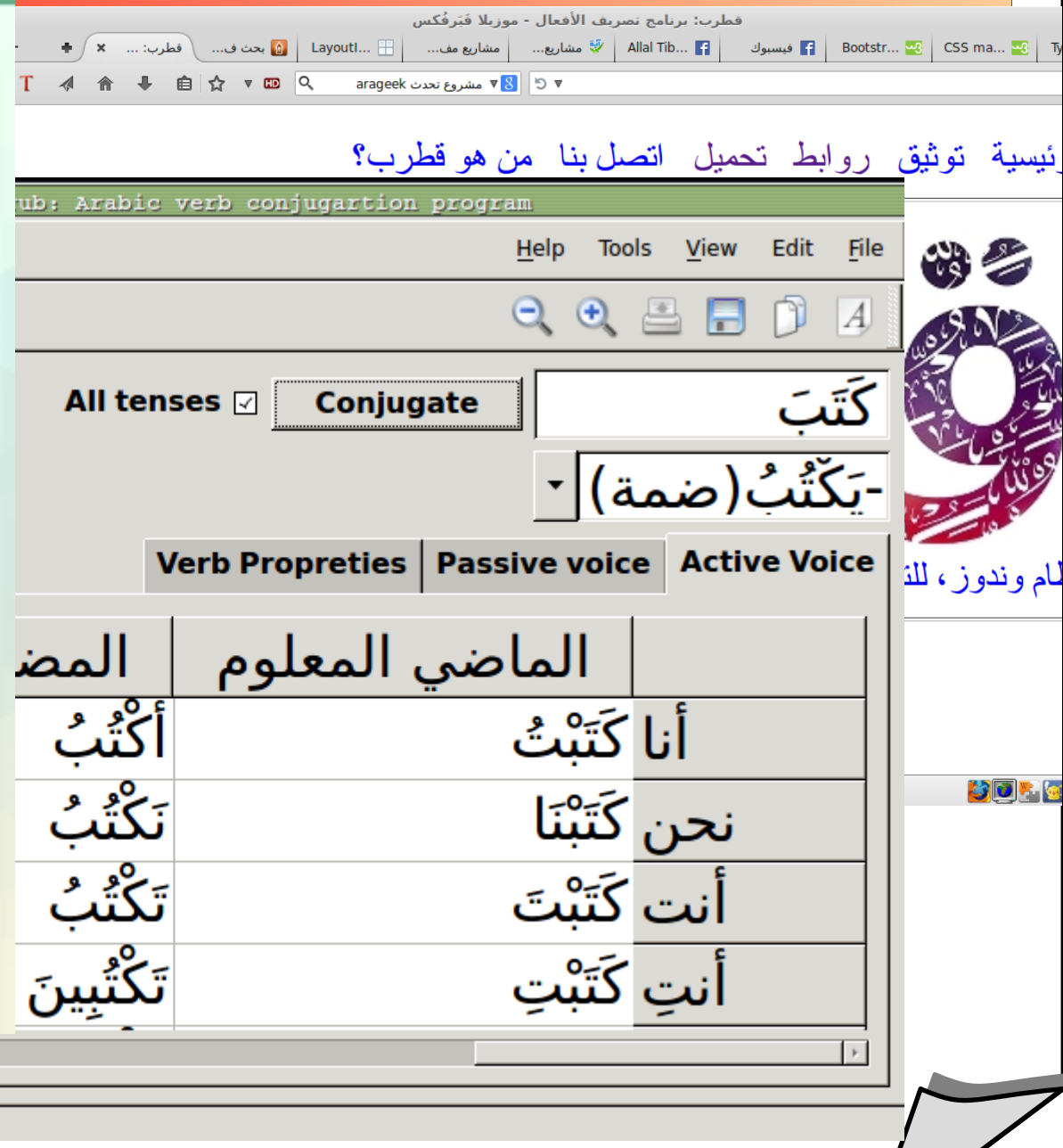
```
def controleur():
    # lire des données a partir du clavier

    data_model =  model()
    affichage = view()

    """  rechercher un nom """
    #
    nom = affichage.input()

    personnes = data_model.rechercher(nom)

    affichage.output(personnes)
```

# Exemple Qutrub



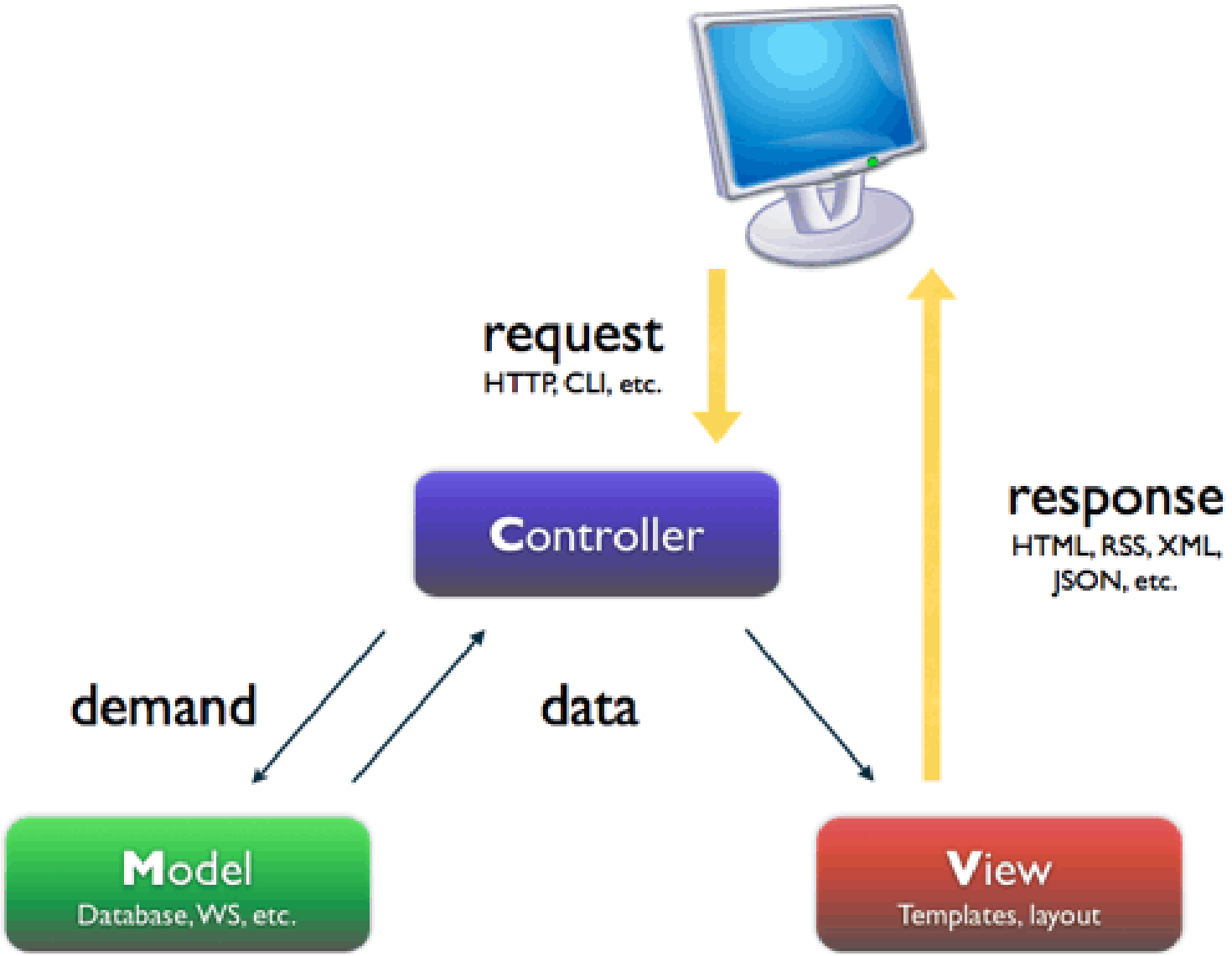
# Exa


04:54
قطـربي
إبـدأ
إلـعب معـنا
خـروج


قطرب: برنامج تصريف الأفعال - موزيلا فَيَرفُكس
من هو قطرب؟ اتصل بنا تحميل روابط توثيق
ub: Arabic verb conjugartion program
Help Tools View Edit File
All tenses
Conjugate
كَتَبَ
-يَكْتُبُ (ضمة)
Verb Propreties
Passive voice
Active Voice
الماضي المعلوم
أنا كَتَبْتُ أكْتُبُ
نحن كَتَبْنَا نَكْتُبُ
أنت كَتَبْتَ تَكْتُبُ
أنتِ كَتَبْتِ تَكْتُبِينَ

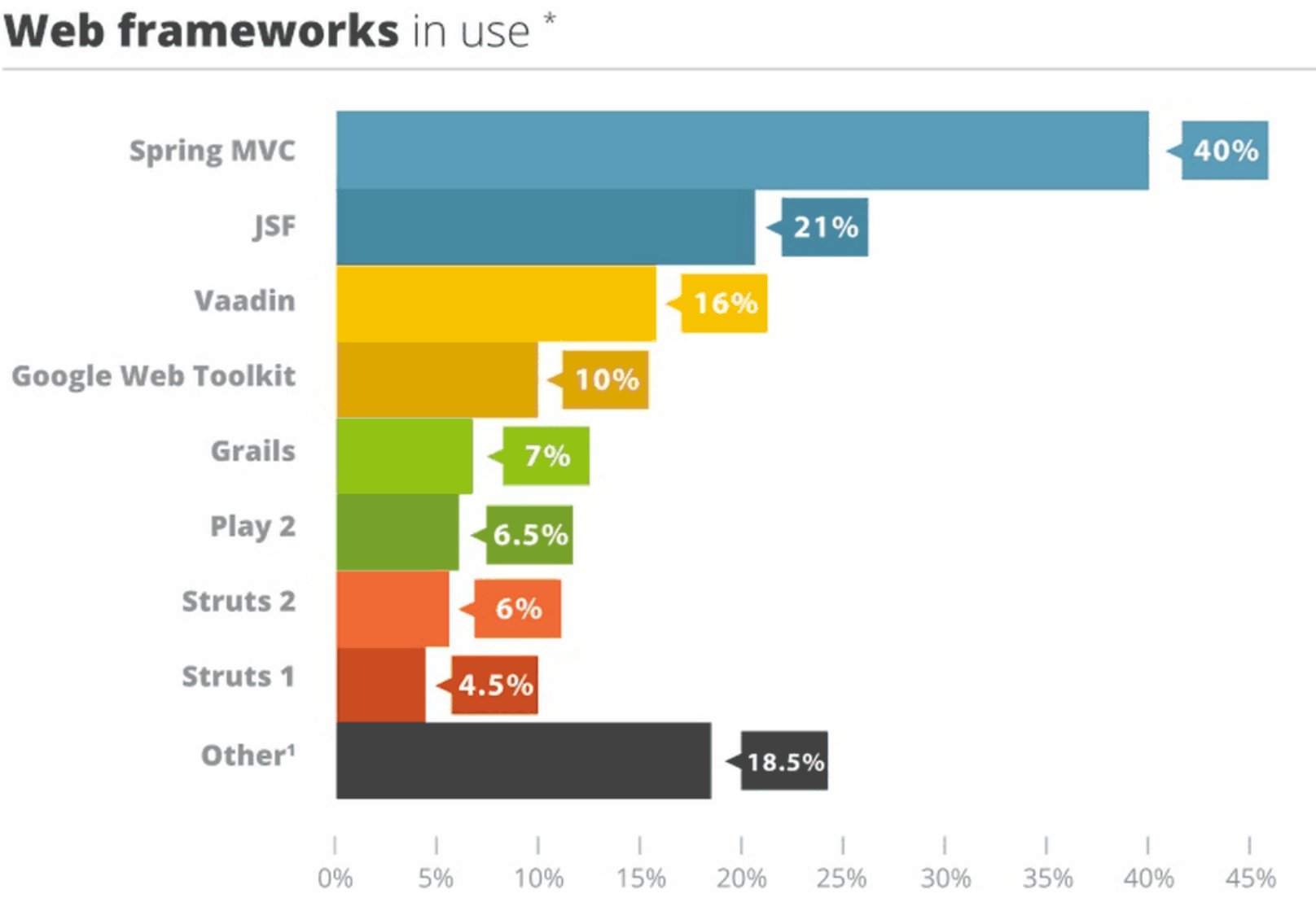
# FrameWork MVC

# MVC


request
HTTP, CLI, etc.
Controller
response
HTML, RSS, XML,
JSON, etc.
demand
data
Model
Database, WS, etc.
View
Templates, layout

# Best 4 Top MVC Java-web-framework

- ht ks-rev
  ea
-



# PHP MVC frameworks





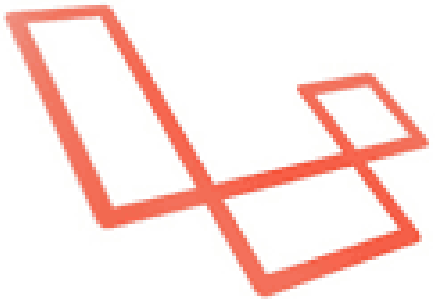


# PHP MVC frameworks









# Python MVC frameworks





Pyramid™