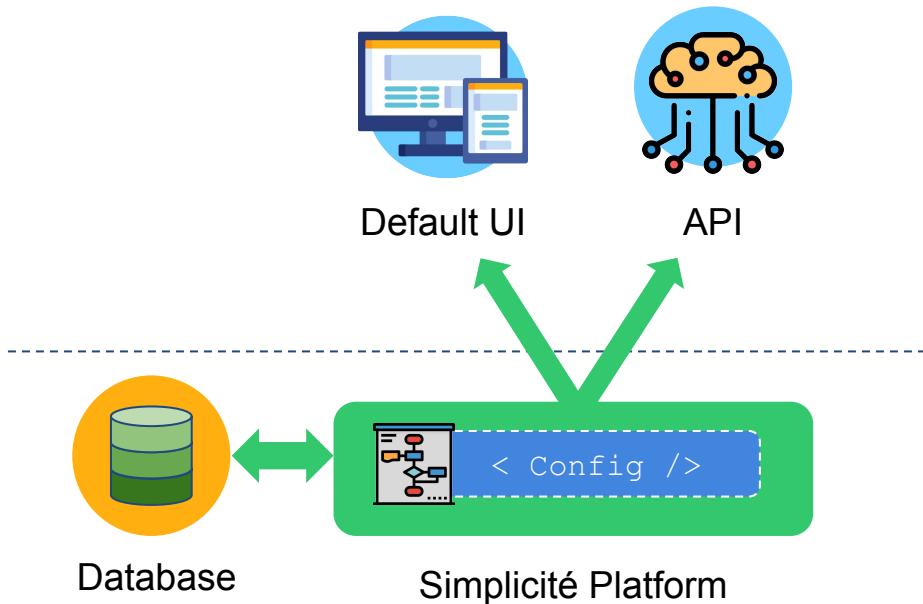
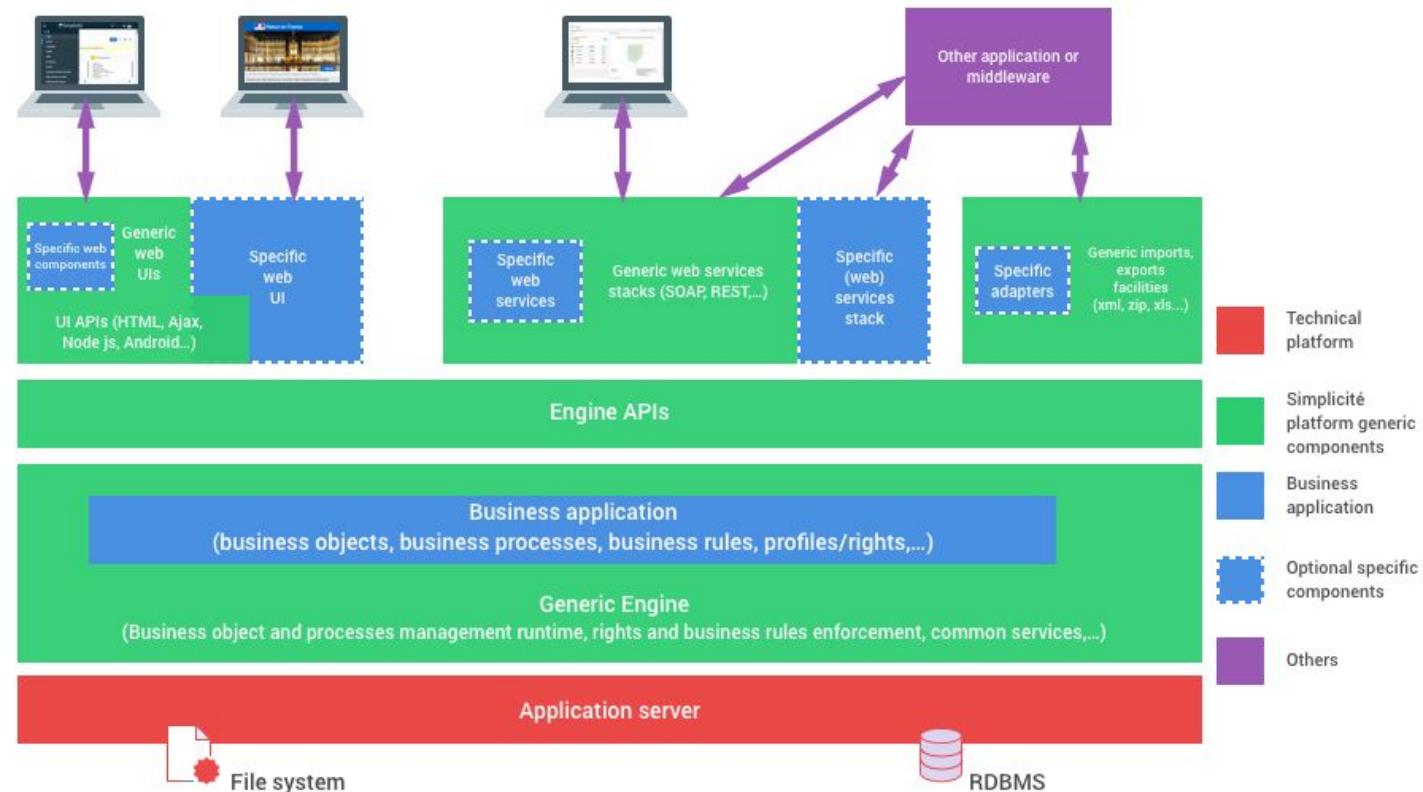


# Une plateforme low-code

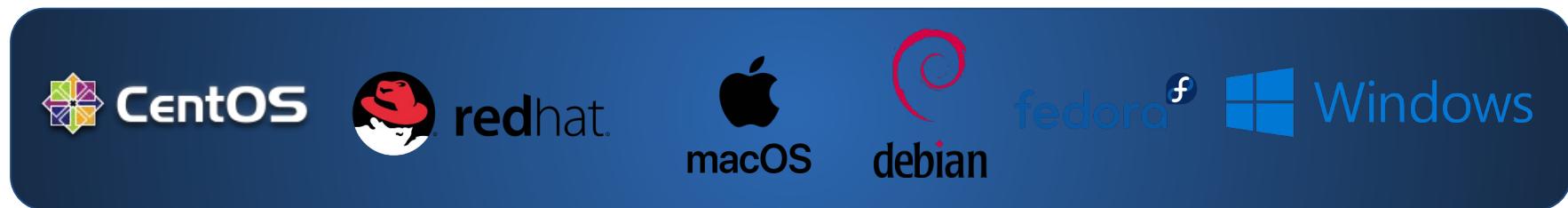


*Simplicité est une plateforme conçue pour aider les ingénieurs à construire efficacement des webapps maintenables en mettant le modèle de données et la logique métier au cœur du processus de construction.*

# Architecture



# Infrastructure nécessaire



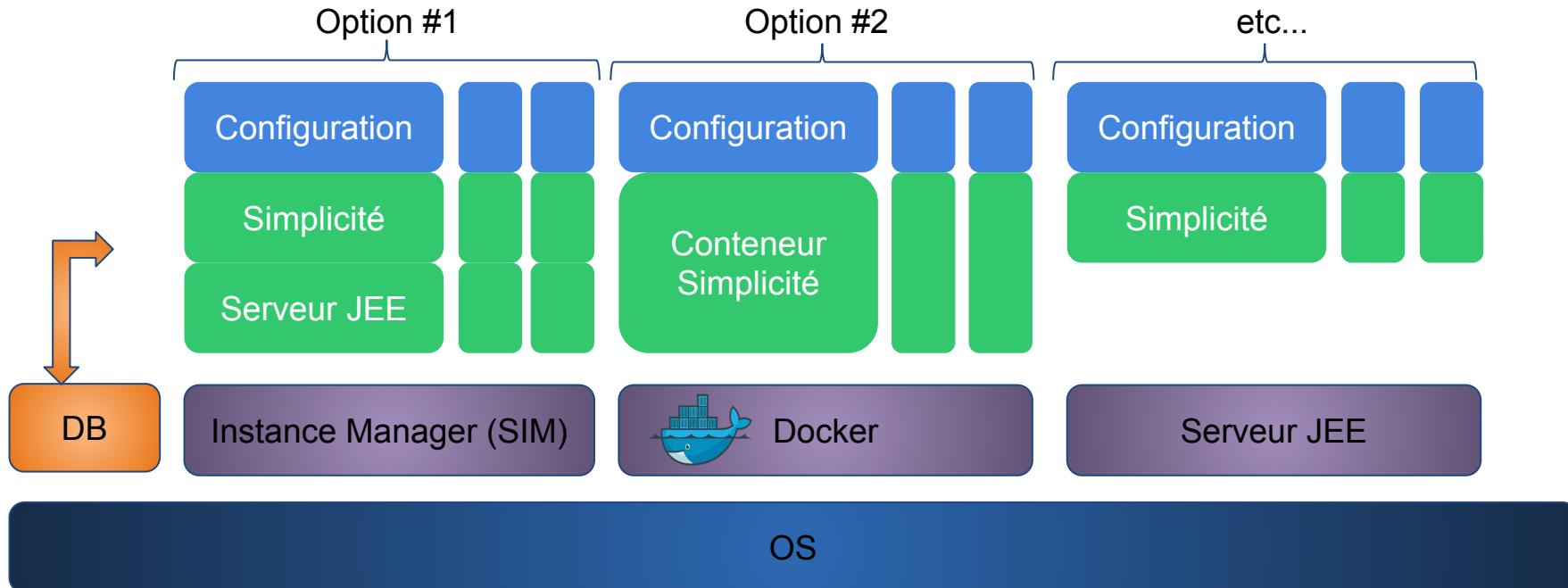
<https://www.simplicite.io/resources/documentation/compliance.md>



# Architectures possibles



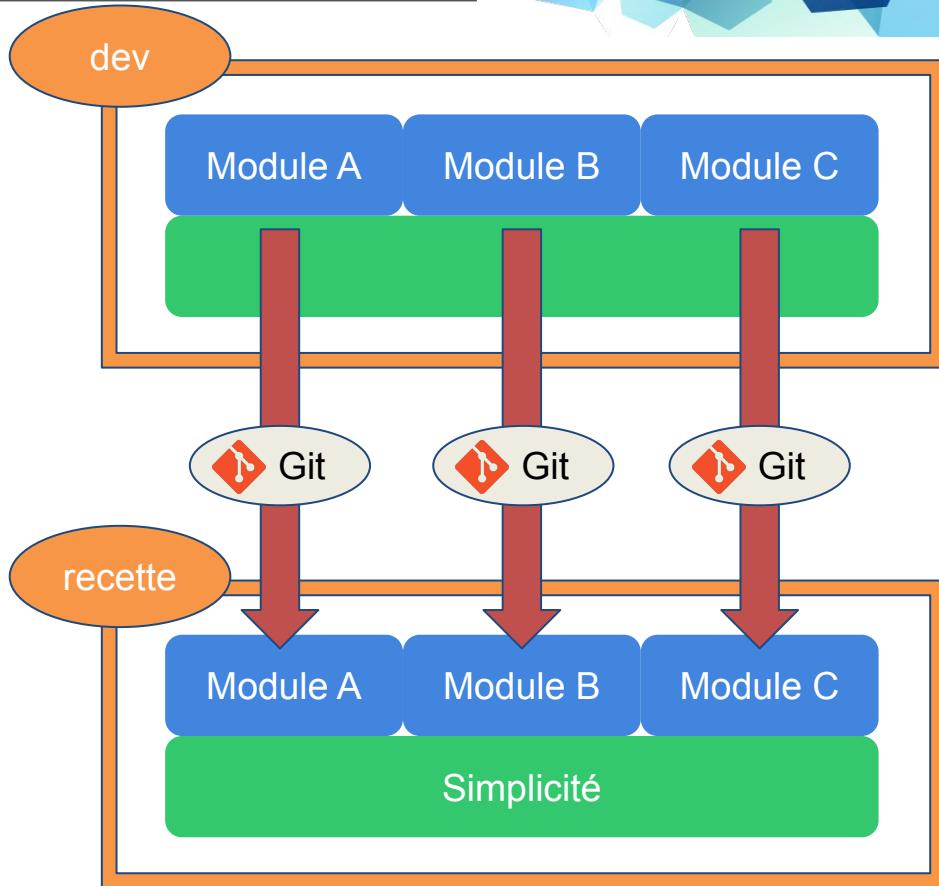
1 instance = JEE + Simplicité + Configuration



# Instance, Application, Module

- **Module:** regroupement d'éléments de paramétrage. C'est l'unité de configuration dans Simplicité.
- **Application:** une application métier peut être composée:
  - d'un module unique
  - de plusieurs modules interdépendants
- **Instance:** une instance Simplicité peut contenir:
  - une application unique
  - plusieurs applications (*déconseillé*)
- **Serveur d'application JEE:** un serveur d'application peut servir une ou plusieurs instances

Conclusion: la division en modules et la répartition entre instances est un choix de conception et d'architecture.

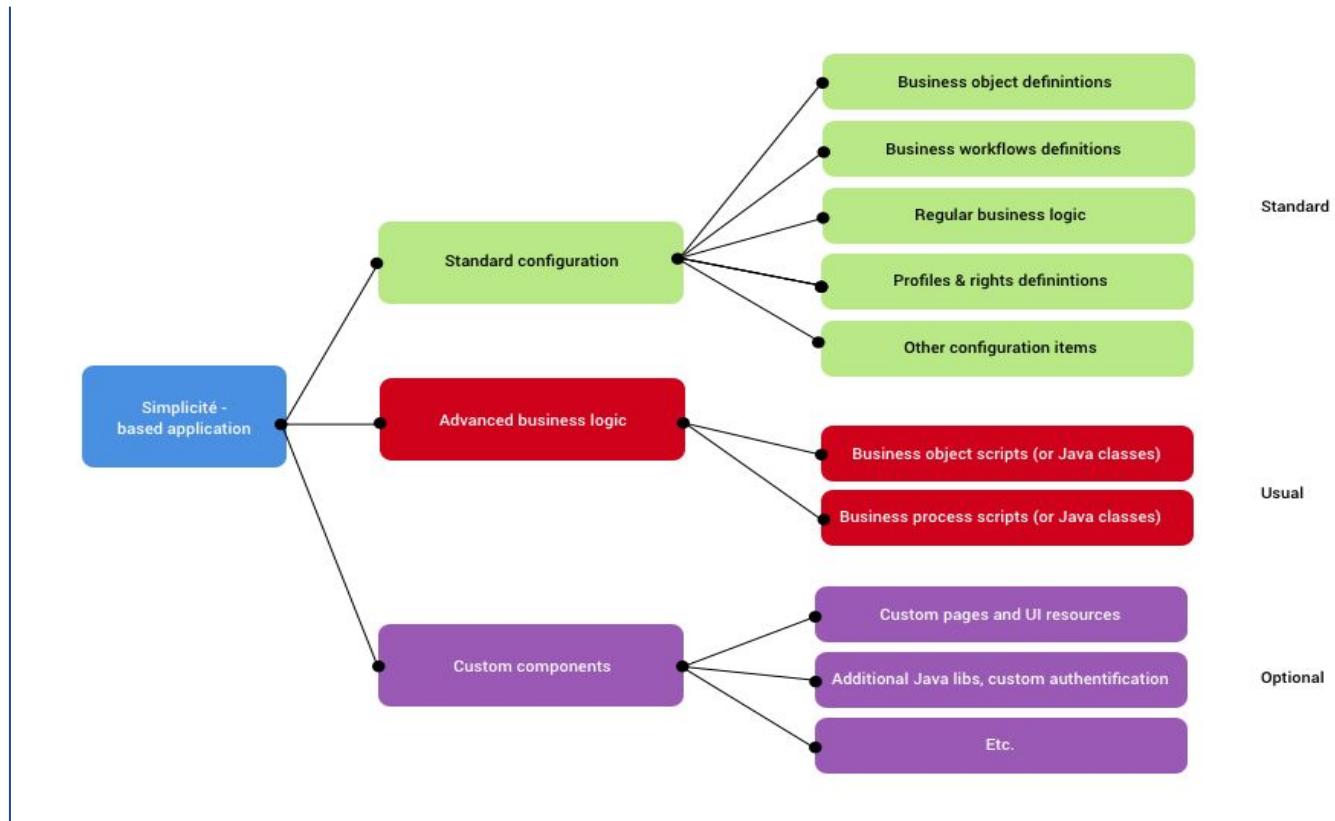
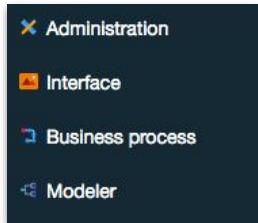
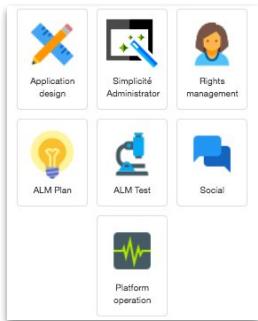


# Composants d'une application Simplicité



## Éléments accessibles via

1. Scopes: "casquettes"
2. Domaines: "menu" (à passer en revue)

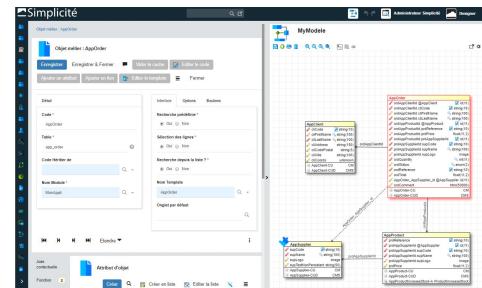


# Outils de configuration

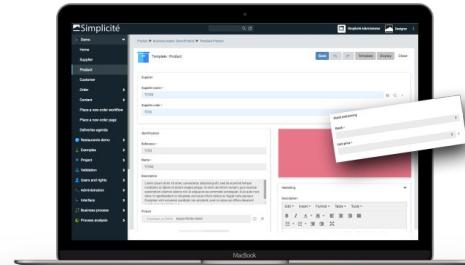
## IHM générique

## Workflows

## Modeler graphique



## Template editor



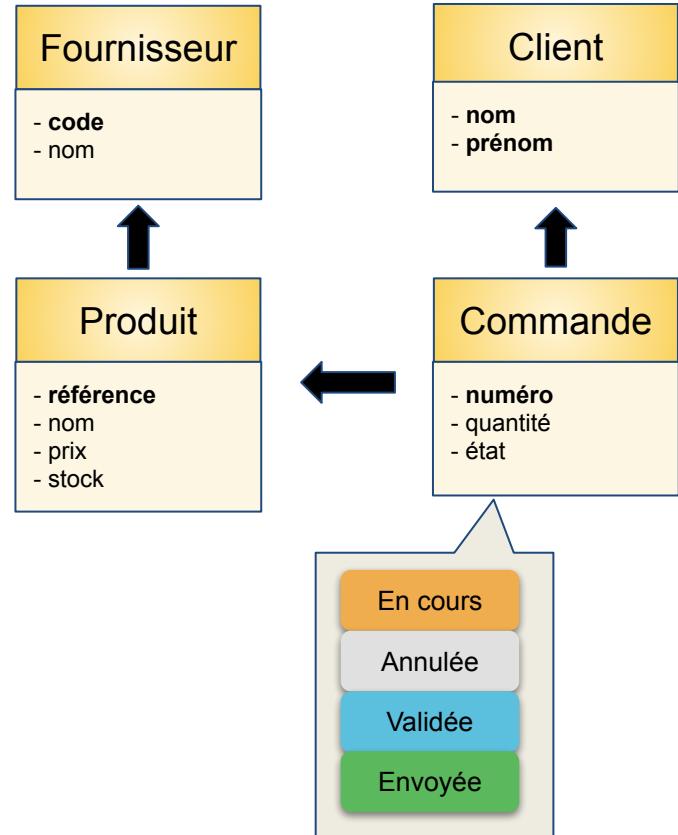
# Cahier des charges

## I - Obligatoire :

1. Des clients passent des commandes de produits, produits par des fournisseurs
2. Les commandes doivent
  - Ne plus être modifiables une fois envoyées
  - Indiquer un prix total en fonction de la quantité et du prix du produit
  - Impacter le stock du produit
3. Une page d'accueil doit récapituler
  - Les commandes en cours
  - Un tableau croisé des commandes
    - par produit
    - Par état
4. On doit pouvoir accéder à la liste des clients d'un fournisseur

## II - Facultatif :

5. Proposer une navigation arborescente
6. Proposer une vue calendaire
7. Proposer une navigation visuelle

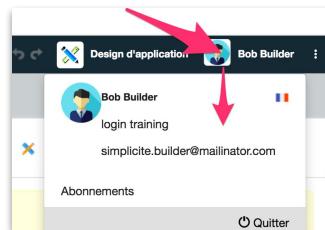




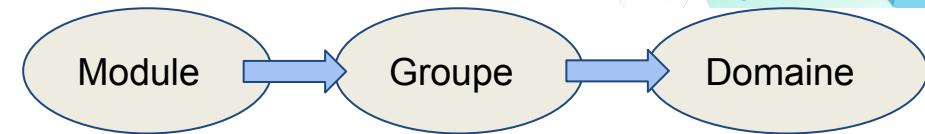
# Principes du paramétrage :

## Création du module

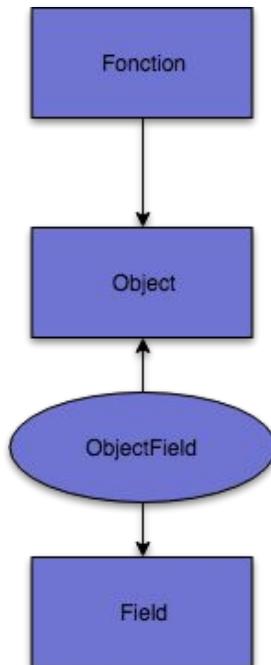
1. Lancer le processus de création de module pour générer:
  - Un module **Training**
  - Un groupe **Superadmin**
  - Un domaine **Menu**
  - Un scope
2. Limiter la visibilité de module
3. Attribuer le groupe nouvellement créé à votre utilisateur



4. Vider le cache de l'application:
  - Alt + C + S



Notions : <https://training.simplicite.io/training/getting-started/>



L'objet métier, centre de l'univers Simplicité.

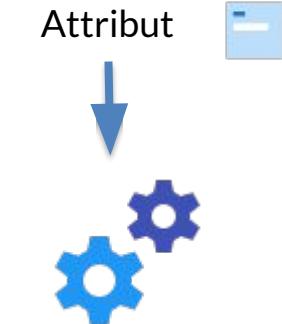
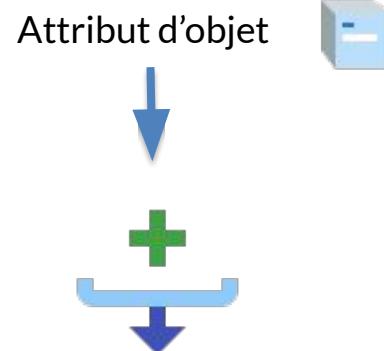
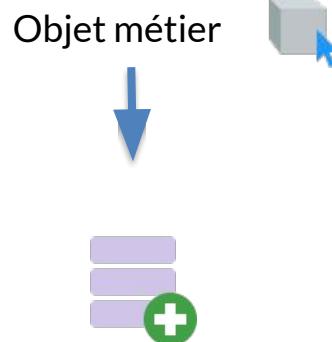
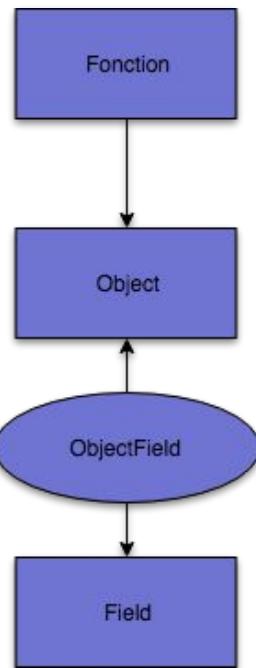
*L'objectif de la formation est de connaître les objets de configuration, leur utilisation, et la relation des uns avec les autres. C'est ce qu'on appelle le méta-modèle.*

*Ce méta-modèle va s'enrichir au fur et à mesure de la formation.*



# Principes du paramétrage :

Objet métier / Attribut d'objet / Attribut

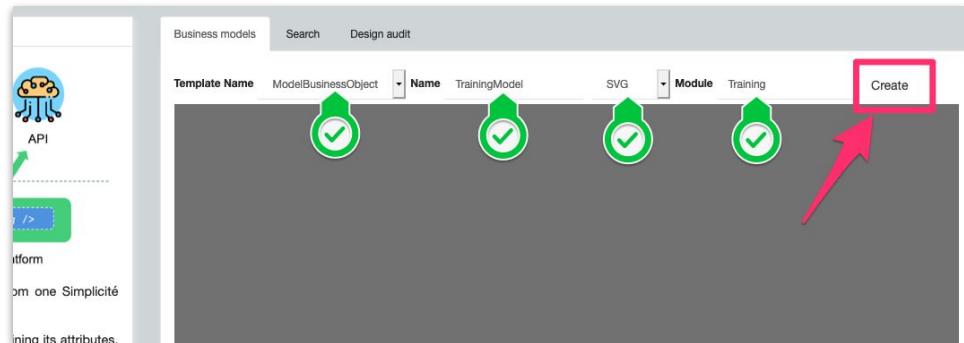


*La notion d'attribut d'objet permet une relation N:N entre l'objet et l'attribut. L'attribut peut ainsi être réutilisé par plusieurs objets (utile par exemple si le client souhaite faire augmenter la taille de tous les champs "description" de 200 à 1000 caractères).*

# Création du modèle

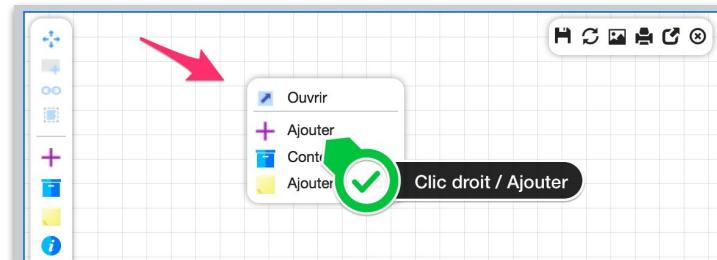
## 1. Créer un nouveau modèle de type **ModelBusinessObject**

Les modèles sont toujours accessibles via le raccourci “sélecteur de modèles” en haut à droite de l'écran.

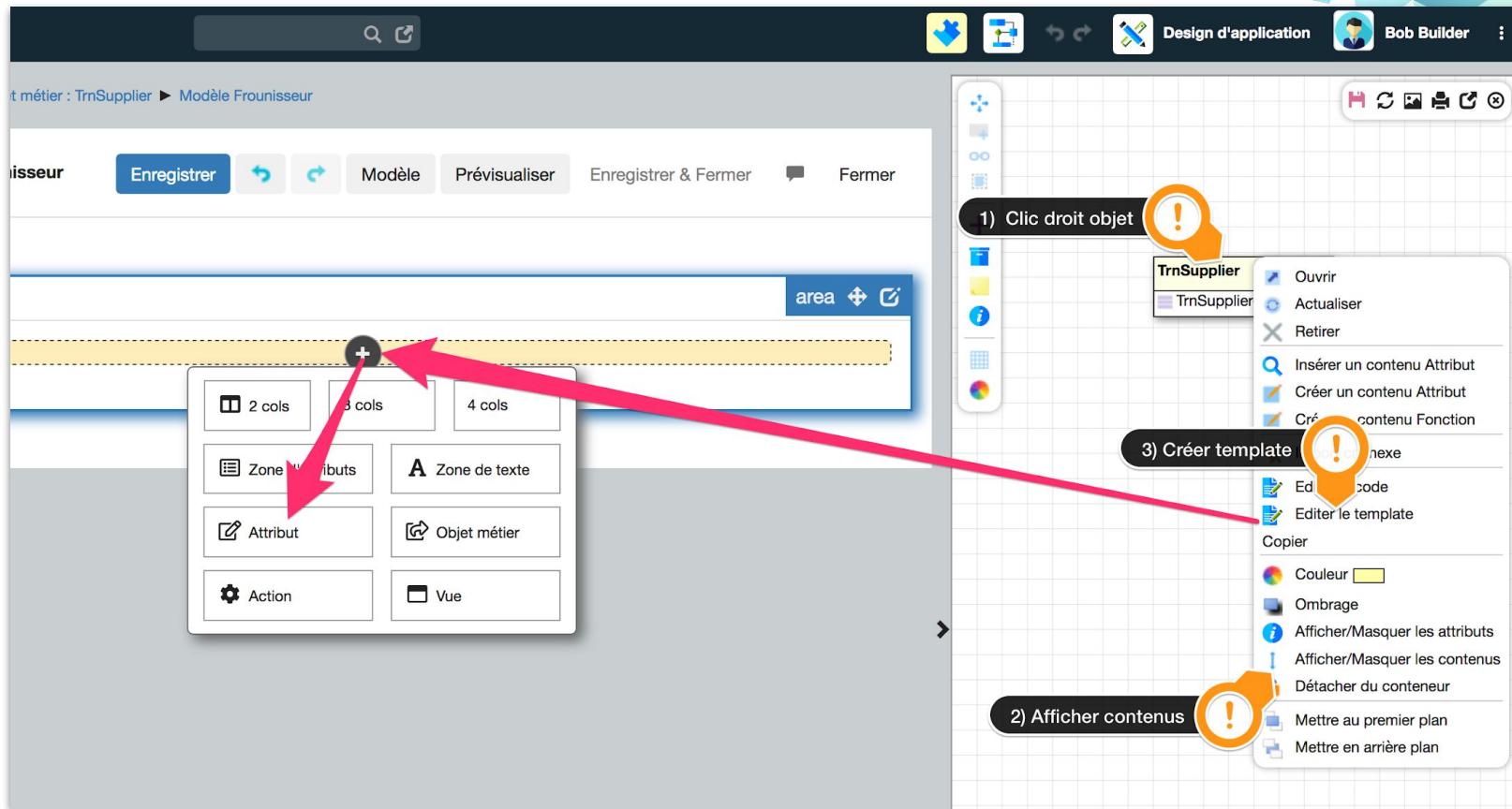


## 2. Lancer le processus de création d'objet métier depuis le modeler

- Clic droit sur le modèle > Ajouter objet “Supplier”



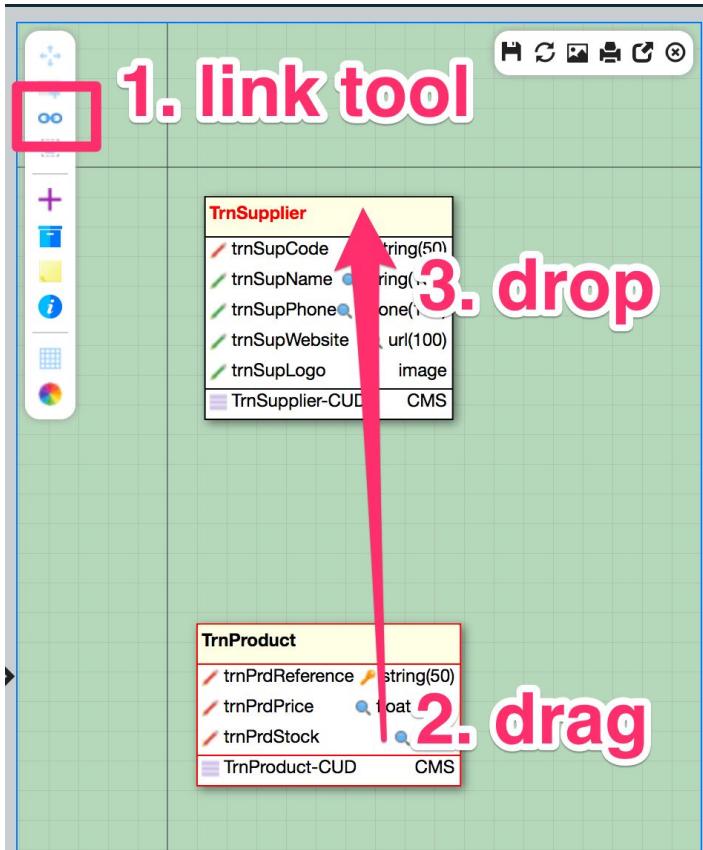
# Création d'un attribut via le template editor



# Exercice : Paramétrage des objets métiers (1/3)

- Créer dans un premier temps les objets et leurs attributs propres (les relations viendront dans un exercice suivant)
- Suivre l'impact sur la base de données (`row_id`)

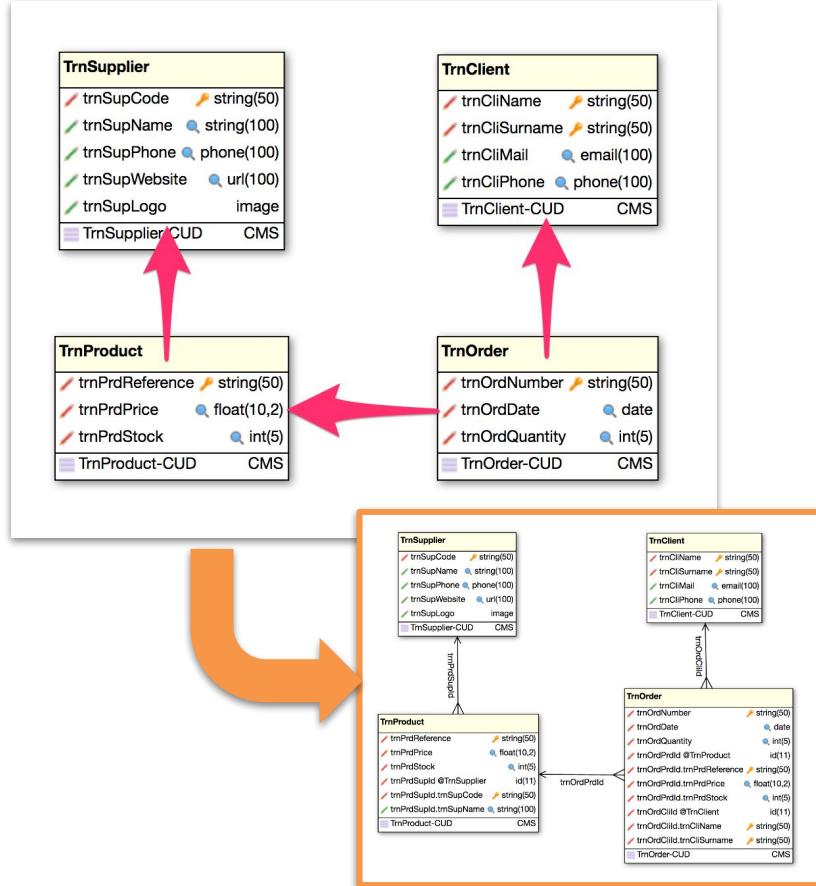
	Fournisseur	Produit	Client	Commande
Obligatoires	<b>code</b>	<b>référence</b>	<b>nom</b>	<b>numéro</b>
		prix	<b>prénom</b>	quantité
		stock		
Facultatifs	nom	nom	mail	date
	téléphone	description	téléphone	
	logo	photo	adresse	
	site			



## Notions:

- Attributs propres VS attributs ramenés
  - via process
  - via template editor
  - à la mano
- Types de suppression:
  - Cascade / null / impossible
  - Cas particulier: ignore
- Cardinalité
- Copie cascade
- Autres options...
- Voir l'impact sur l'application !

# ∞ Relations 1:N



## Liens:

- Pendant le process de création de lien, on ramène les champs que l'on souhaite afficher sur le formulaire du fils (par exemple la référence et le prix du produit sont intéressants sur le formulaire de la commande).

# Exercice Template + Relations : la commande

Commande : Commande N°289

Enregistrer Enregistrer & Fermer Annuler Fermer

**Identification**

Numéro \* 289

Date 22/02/2018

Etat \* En attente

Date de livraison 25/02/2018 08:00:00

**Commande**

Prix unitaire 560,00 €

Quantité \* 2

Total commande 1 120,00 €

TVA 224,00 €

**Client**

Code client \* CLI002

Prénom client \* Robert

Nom client \* SPONJE

**Produit**

Référence produit \* REF009

Nom produit \* Tablet PC

Code fournisseur \* DY

Nom fournisseur \* Dyewlett Yackard

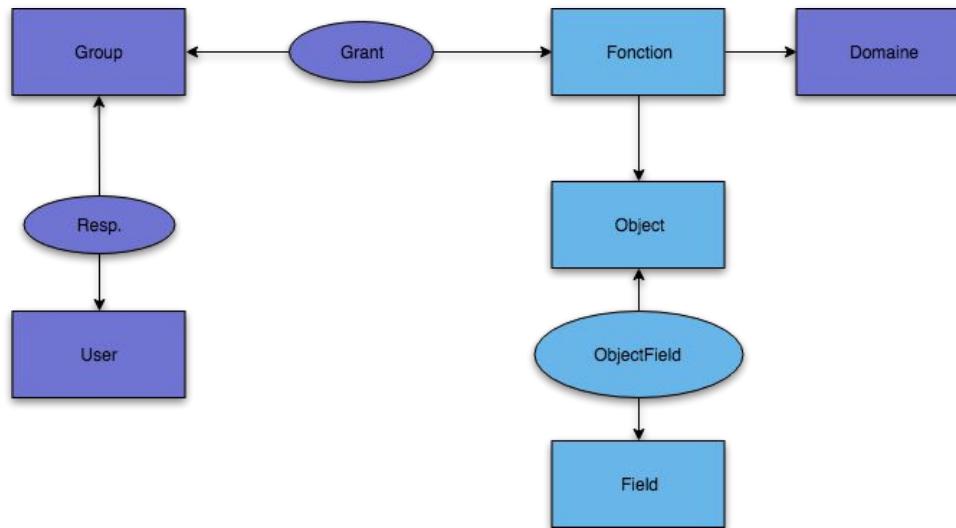
Stock Produit: \* 98

Prix unitaire courant \* 560,00 €

Details >

Back Next

# Gestion des droits : Méta-Modèle

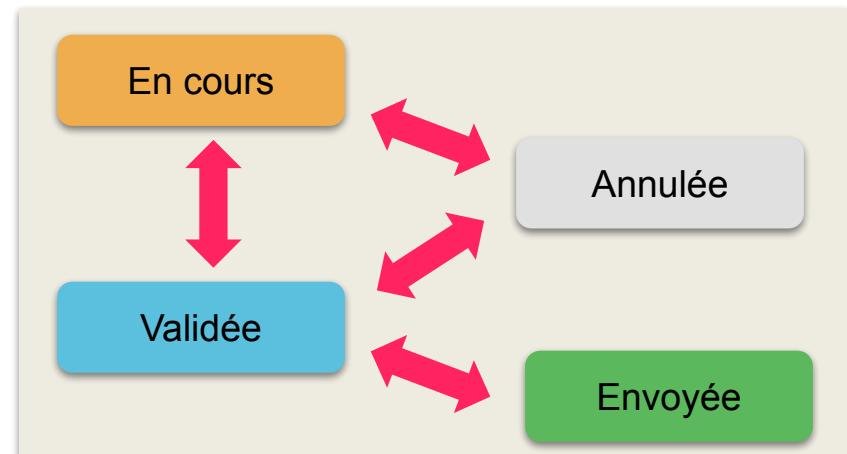
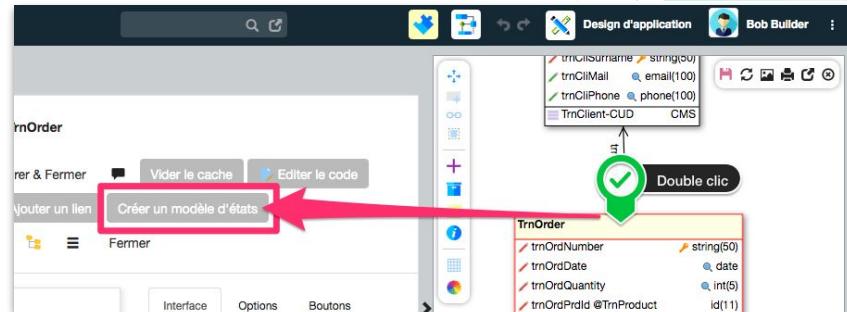


Exercice: Implémenter les droits sur les objets

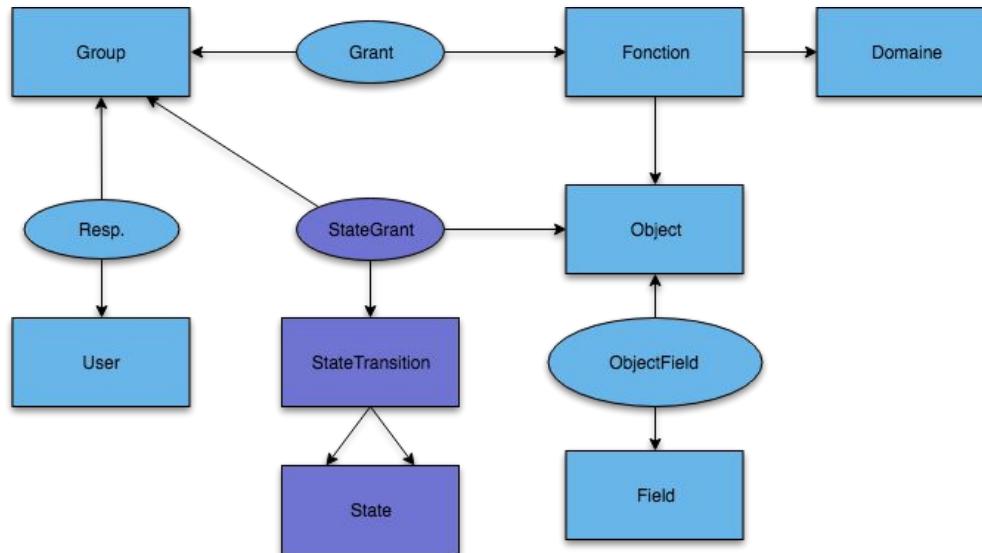
# Diagramme d'état

## Création :

1. [prérequis] créer un attribut “état” sur la commande:
  - o type “énuméré simple”
  - o obligatoire
2. À partir de la définition de l’objet, lancer le process “Créer un modèle d’états”
3. Paramétriser les transitions d’états possibles, leurs traductions, leurs droits. *Penser aux retours arrière.*



# Transitions d'états : Méta-Modèle





# Création de styles d'attributs



## Menu: Interface / Style d'attribut

Exercice : créer un style d'attribut pour associer un fond de couleur à l'attribut statut de la commande.

Commandes

[Créer](#) [Créer en liste](#) [Editer la liste](#) [Agenda](#) [XML publication exemple](#) 100

Les commandes sont ordonnées par défaut par numéro de commande décroissant

Identification			Client		Produit			Commande		
<input type="checkbox"/>	Numéro	Date	Etat	Code client	Référence produit	Nom fournisseur	Quantité	Total commande	TVA	
<input type="checkbox"/>	254	25/04/2019	Validée	CLI001	REF003	BIM Computers Ltd	1	880,00 €	176,00 €	
<input type="checkbox"/>	253	25/04/2019	En attente	CLI001	REF009	Dyewlett Yackard"	1	519,00 €	103,80 €	
<input type="checkbox"/>	252	25/04/2019	En attente	CLI003	REF009	Dyewlett Yackard"	2	1 038,00 €	207,60 €	
<input type="checkbox"/>	251	25/04/2019	Envoyée	CLI003	REF006	LLED Computers	4	2 220,00 €	444,00 €	
<input type="checkbox"/>	250	24/04/2019	En attente	CLI001	REF005	BIM Computers Ltd	1	150,00 €	30,00 €	
<input type="checkbox"/>	249	24/04/2019	En attente	CLI001	REF009	Dyewlett Yackard"	3	1 557,00 €	311,40 €	
<input type="checkbox"/>	248	16/04/2019	Annulée	CLI004	REF005	BIM Computers Ltd	2	300,00 €	60,00 €	

# Champs Calculés

Le champ `ordTotal`, déjà créé, doit être configuré en tant que **champ calculé**

- En lecture seule
- Avec une formule de calcul :

```
[VALUE:appOrdPrdId.appOrdP  
rice] * [VALUE:ordQuantity]
```

The screenshot shows the 'Simplicité' application's configuration interface for a calculated field. On the left, there is a sidebar with a tree view of objects: Gestion des commandes, Accueil, Fournisseur, Produits, Client, Commande (selected), Tout voir, Bannières, En cours, Annulé, Livré, Validé, Métriques, App product avec prix, test, AppOrderExt, Référence, Pizzeria, and Périmètre. The main area is titled 'Simplicité' and shows the configuration for a field named 'ordTotal'. The 'Type' is set to 'Décimal (double)' and 'Type validation' is 'Liste de valeurs' with a value of '11'. The 'Précision' is '2' and 'Taille minimale' is '0'. The 'Expression' tab contains the formula: `[VALUE:ordAppProductId.prdPrice]*[VALUE:ordQuantity]`. The right side of the screen shows various configuration options: Auto-complétion (Oui), Enregistrer, Enregistrer & Fermer, Fermer, Affichage, Visible (Partout), Etendu ? (Non), Recherche possible (Oui), Etendu en liste ? (Non), Recherche obligatoire (Non obligatoire), Casse, Présentation (Droite à gauche), and a 'Réduire' button.

Exercice :

Créer un champ calculé égal au montant de la TVA (`ordTVA`) via l'utilisation d'un paramètre système (`APP_TVA`)

```
[VALUE:ordTotal] * ([SYSPARAM:APP_TVA]/100)
```

# { } Les règles de gestion via contraintes

---

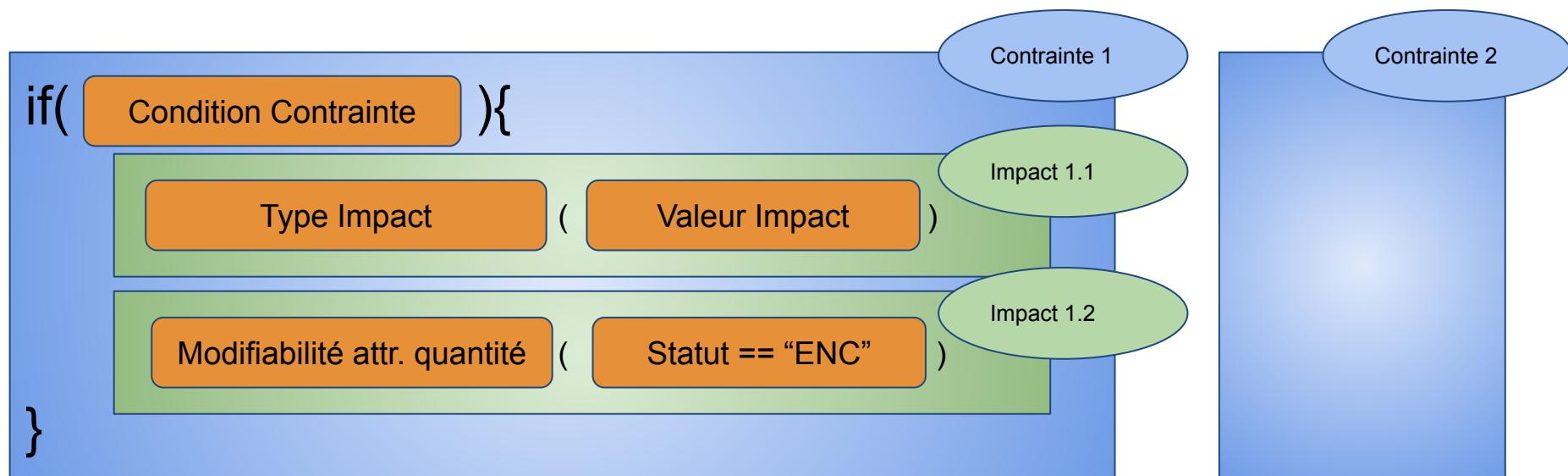


Une contrainte est composée d'éléments permettant de contrôler l'exécution d'une règle de gestion

- Une **condition d'exécution** de la contrainte (attribut booléen OU résultat d'une méthode OU Expression)
- Des impacts sur les propriétés :
  - d'un **attribut** (modifiable, visible, obligatoire, etc.)
  - d'une **action, vue ou zone d'attributs** (visibilité)
  - de l'**objet métier** (copier, créer, modifier, etc.)

Les éléments de la contrainte génèrent du code qui est exécuté au cours du cycle de vie de l'objet.

L'ordre des contraintes et des impacts a son importance: étant exécutés dans cet ordre, un impact peut en écraser un autre.



# { } Les règles de gestion via contraintes

Contrainte : AppOrder-C1

Enregistrer Enregistrer & Fermer Editer la matrice Fermer

Contrainte

Nom \* AppOrder-C1

Code Objet \* AppOrder

Nom Module \* MonAppli

Ordre \* 1

Effets \*  Statique  Front-end  Back-end

Condition

Type \* Expression

Expression \* 

i 1	true
-----	------

Méthode

Nom logique Attribut booléen

Etendre ▾

# { } Exemple de Contraintes

- La quantité de la commande ne peut pas être modifiée lorsque la commande est en statut validé :
  - Ajouter une contrainte sur l'objet AppOrder comme condition `true` (toujours applicable)
  - Ajouter un impact sur la contrainte sur le champ `appOrdQuantity` avec comme expression :  
`[ISNEW] || [STATUS].equals("ENC")`
  - Ajouter un impact sur la contrainte sur le champ `appOrdReference` avec comme expression: `[ISNEW] || [STATUS].equals("ENC")`
- Cette règle peut être limitée par Groupe.
- Les nouvelles commandes doivent aussi être testées car leur statut est vide par défaut
- L'ordre des impacts et des contraintes est important.



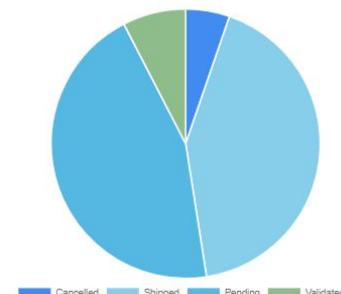
# Tableaux croisés



## Exercice

- Créer un tableau croisé qui synthétise l'état des commandes
  - Axe verticaux
    - i.Fournisseur
    - ii.Produit
  - Axe horizontal
    - i.État
  - Axe “valeur” : aucun (=> dénombrement), mais ce pourrait également être le prix pour une synthèse financière
- Permettre l'affichage d'un graphique

	Fournisseur A		Fournisseur B	
	Produit A1	Produit A2	Produit B1	Produit B2
En cours	#	#	#	#
Annulé	#	#	#	#
Validé	#	#	#	#
Envoyé	#	#	#	#



# Historisation

---



1. Option “Data history” sur l’objet métier
2. Ajout de l’objet généré au diagramme (AppProductHistoric)
3. Supprimer les attributs qu’on ne souhaite pas prendre en compte
4. Attribuer les droits sur l’objet
5. Clear cache



# Création de vues



The screenshot displays a web application interface with several distinct sections:

- Pending orders:** A table showing three pending orders with columns for Identification, Date, Status, Customer code, Customer address 1, Product reference, Supplier name, Quantity, Total, and VAT.
- Orders:** A large image of a pizza with toppings like olives, onions, and pepperoni, labeled "Pizzeria". Below it are three smaller images of pizzas labeled "Beef", "Margherita", and "Pepperoni".
- Monitoring:** A dashboard with various metrics:
  - System:** Shows Java heap usage (Max heap: 488432 Kbytes, Free heap: 82095 Kbytes, Allocated: 291940 Kbytes, Peak: 286771 Kbytes).
  - Sessions:** Shows session statistics (Public: 0, Private: 1, Peak: 1, Max sessions: 1, Max time: 0:29:55).
  - Cache:** Shows cache statistics (Objects: 96 / 10000, Direct: 10 / 9, Processes: 1 / 10000).
- Monitoring (Details):** A detailed monitoring section titled "Monitoring" with tabs for Snapshot, History, Data, Thread, Perf, Agents, and Dumps. It shows real-time data for various metrics like Object, Thread, Process, and Memory usage.
- Memory Dump:** A detailed memory dump showing memory usage over time (15/03/2018) with charts for Compressed Data Size, RI Data Size, PS Data Size, and GC Data Size. It also provides statistics for Public, Private, and Peak sessions, along with heap details like Max heap (495 Mo), Free heap (250 Mo), Allocated (395 Mo), and Objects (97 / 10000).

Une vue peut se définir pour devenir :

- la page d'accueil de l'application
- l'accueil d'un domaine fonctionnel
- L'accueil d'un groupe
- une vue spécifique : la page pourra être utilisée par accès via un objet externe



# Recherche prédefinie

Rechercher Commande

Recherche globale

Code

Prénom

Nom

Référence

Code

Nom

Quantité

Etat

En cours  
 Annulé  
 Livré  
 Validé  
 (vide)  
 (non vide)

Référence

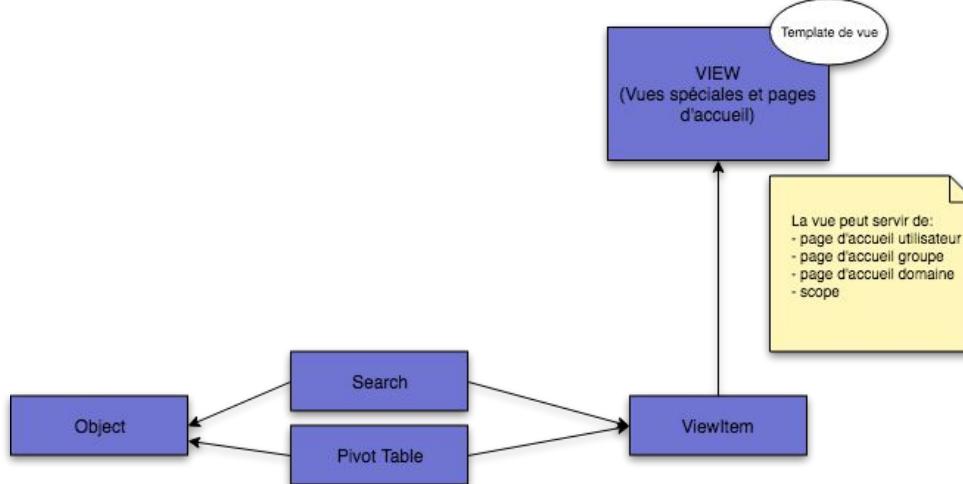
Rechercher      Annuler      Effacer      Fermer

Recherche prédefinie >

- Tous les utilisateurs peuvent créer des recherches prédefinies.
- Une méthode simple consiste à se positionner sur le moteur de recherche de l'objet dont on veut effectuer une recherche prédefinie , saisir ces critères et tris et lui donner un nom

Exercice : créer une recherche simple sur les commandes en attente

# Méta-Modèle: Vue / TC / Recherche



## Exercice :

Créer une vue AppHome qui sera la page d'accueil du domaine Gestion des commandes, qui contiendra 2 zones

- Zone 1 : tableau croisé AppOrder-TC1,
- Zone 2 : La liste des messages partagés

Affecter cette vue à un utilisateur



# Création de raccourcis



La liste présente tous les raccourcis triés par ordre

La barre de raccourcis présente ceux qui sont habilités à l'utilisateur

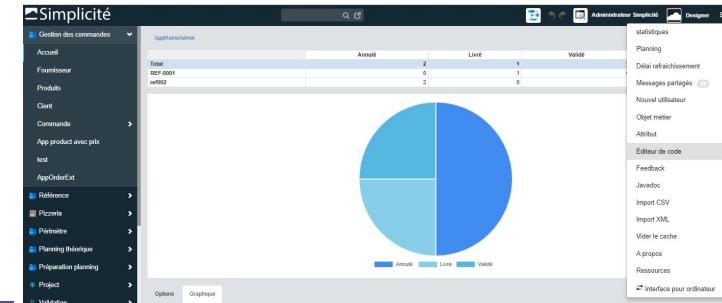
Un raccourci est un lien URL interne ou externe, une zone de destination.

Exemple d'url interne:

- [EXPR:HTMLTool.getListURL("AppClient", true)] : liste des clients
- [EXPR:HTMLTool.getFormURL("AppClient", null, ObjectField.DEFAULT\_ROW\_ID, null)] : formulaire de création d'un client
- [EXPR:HTMLTool.getProcessStartURL("AppCreerClient ")] : processus de création d'un client

Exercice :

- Créer un raccourci AppCreerClient pour créer un client





# Thème

★ Thème

Nom \* PizzeriaTheme

Thème de base \* Base claire

Logo scope Choisir un fichier Aucun fichier choisi

Nom Module \* Pizzeria

Styles complémentaires Choisir un fichier Aucun ...hoisi Edit

Fichier CSS généré PizzeriaTheme\_gen.css

★ Titre / Pied de page

Entête #D98039

Logo entête pizzeria-logo.png

En pied #D98039

Couleur du texte #FFFFFF

Couleur du texte #FFFFFF

Boîte de recherche #FFFFFF

Logo inversé Oui Non

★ Menu

Domaine #D98039

Texte du domaine #FFFFFF

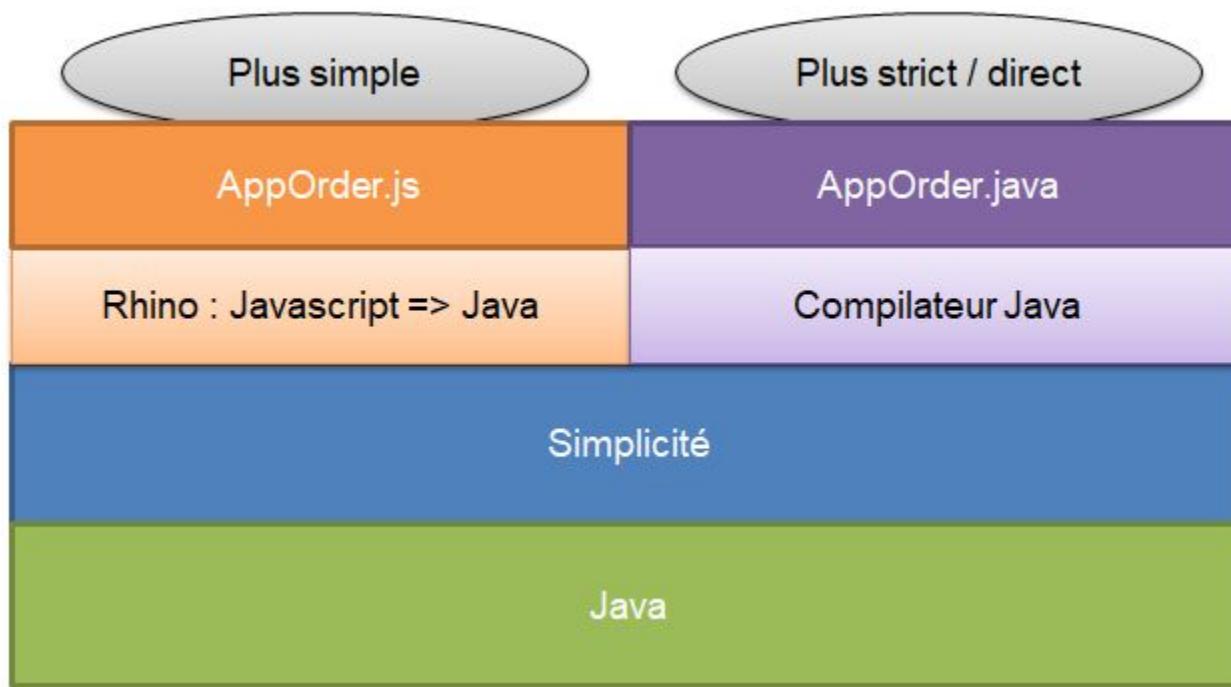
Arrière plan #D98039

★ Accueil



- Développement

# Scripts



# Hooks et cycle de vie de l'objet



- L'objet suit un cycle de vie, la plateforme va le solliciter et effectuer des opérations (*recherche, création etc.*) en fonction des actions de l'utilisateur
- À la plupart des étapes sont liées des "hooks", qui permettent de surcharger le comportement de l'objet avec du code. On en trouve de différents types:
  - Init:
    - Utilisation: préparation de l'objet avant *affichage*
    - Ex: initCreate(), initUpdate(), initCopy()
  - Pré / Post:
    - Utilisation: *règles de gestion* avant / après requête
    - Ex: preCreate(), preValidate(), preSearch(), preUpdate(), preSave(), preDelete()
      - Ex: postCreate(), postValidate(), postSearch(), postUpdate(), postSave(), postDelete()
  - postLoad: appelé une seule fois par chargement de nouvelle instance d'objet
- La liste exhaustive est disponible dans la javadoc, tous les objets héritent de ScriptedObjectDB

# Exemple de Hook : postLoad

- Créer un script sur l'objet AppClient :

```
package com.simplicite.objects.MonAppli;

import java.util.*;
import com.simplicite.util.*;
import com.simplicite.util.tools.*;

public class AppClient extends ObjectDB {

    @Override
    public void postLoad() {
        String log = "AppClient: " + getInstanceName();
        log += " user:" + getGrant().getLogin();
        AppLog.info(getClass(), "postLoad", log, null);
    }

}
```

- Ce code affiche du contenu dans la console et nous permet de voir le fonctionnement du cache.  
Vider le cache de l'objet pour le message de nouveau.

# Un exemple de Hook : postValidate



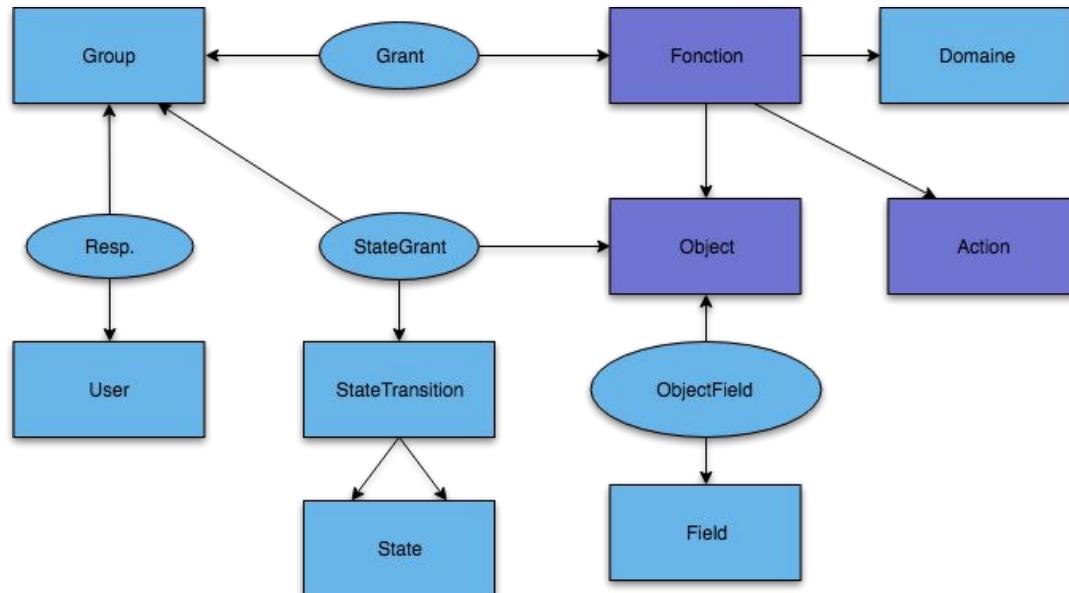
- La quantité d'une commande doit être positive
- Créer un script sur AppOrder avec la logique suivante

```
@Override  
public List<String> postValidate() {  
    List<String> msgs = new ArrayList<String>();  
    if (getField("appOrdQuantity").getInt(0) <= 0){  
        msgs.add(Message.formatError("APP_ERR_QUANTITY", null, "appOrdQuantity"));  
    }  
    return msgs;  
}
```

- Sur la même logique, ajouter un script qui vérifie que le stock n'est pas inférieur à la quantité du produit commandé.

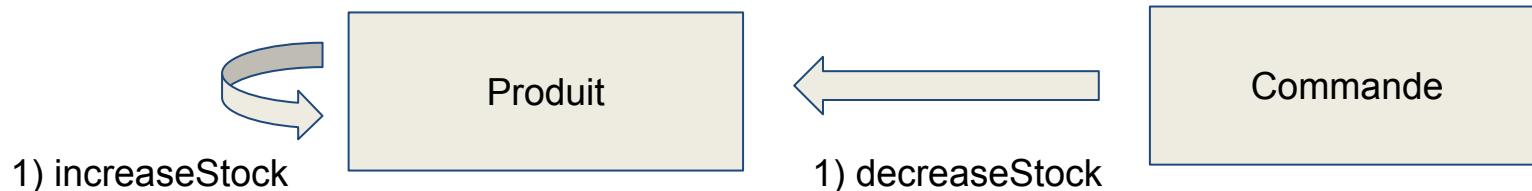


# Actions : Méta-Modèle





# Exercice: Actions



1. Une action simple sur le produit permettant augmenter son stock.
2. Une action avancée permettant de diminuer le stock du produit lorsqu'une commande est passée
  - o Utilise un paramètre d'objet
  - o Appelle une action d'un autre objet



# Action simple



- Créer une action de type méthode sur l'objet AppProduct
- Créer un script sur l'objet AppProduct qui incrémente le stock du produit :

```
package com.simplicite.objects.MyApplication;

import java.util.*;
import com.simplicite.util.*;
import com.simplicite.util.tools.*;

/**
 * Business object AppProduct
 */
public class AppProduct extends ObjectDB {
    private static final long serialVersionUID = 1L;

    public void increaseStock(){
        ObjectField prdStock = getField("prdStock");
        prdStock.setValue(prdStock.getInt(0)+10);
        save();
    }
}
```



# Action avancée



- Créer une méthode sur l'objet AppProduct pour décrémenter le stock.

```
public void decreaseStock(int qte){  
    ObjectField prdStock = getField("appPrdStock");  
    prdStock.setValue(prdStock.getInt(0)-qte);  
    save();  
}
```

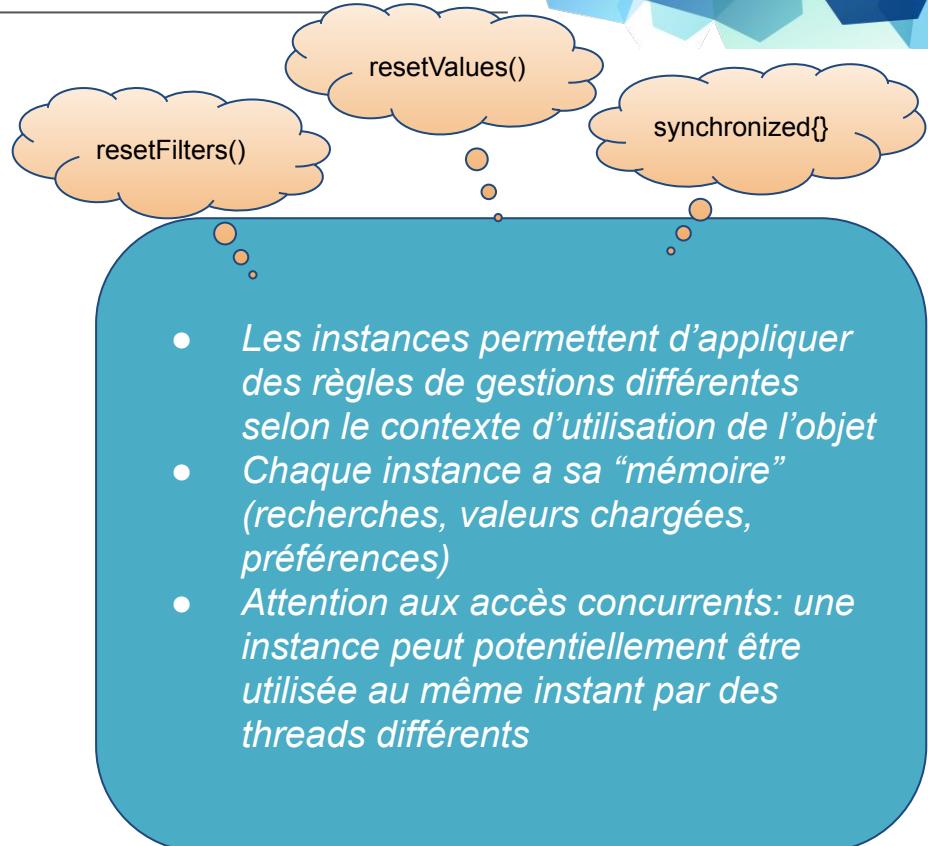
- Appeler la méthode sur la transition d'état Validé de l'objet AppOrder, en utilisant le hook postUpdate.

```
public class AppOrder extends ObjectDB {  
    private static final long serialVersionUID = 1L;  
  
    @Override  
    public String postUpdate() {  
  
        if(getOldStatus().equals("ENC") && getStatus().equals("VAL")){  
            AppProduct prd = (AppProduct) getGrant().getTmpObject("AppProduct");  
            prd.select(getFieldValue("appOrdProductId"));  
            prd.decreaseStock(getField("appOrdQuantity").getInt(0));  
        }  
  
        return null;  
    }  
}
```



# Les instances d'objet

- Les instances génériques
  - Instance Main :
    - Page de recherche
    - Les listes
    - Les formulaires
  - Instance des panels pour les objets liés
  - Instance de référence pour les objets référencés
  - Instance des processus
  - Instance de template de publication, etc, ...
- Autres instances
  - Ajax instance (APIs)
  - Instance(s) Ad hoc, e.g. tmplInstance for specific business logic

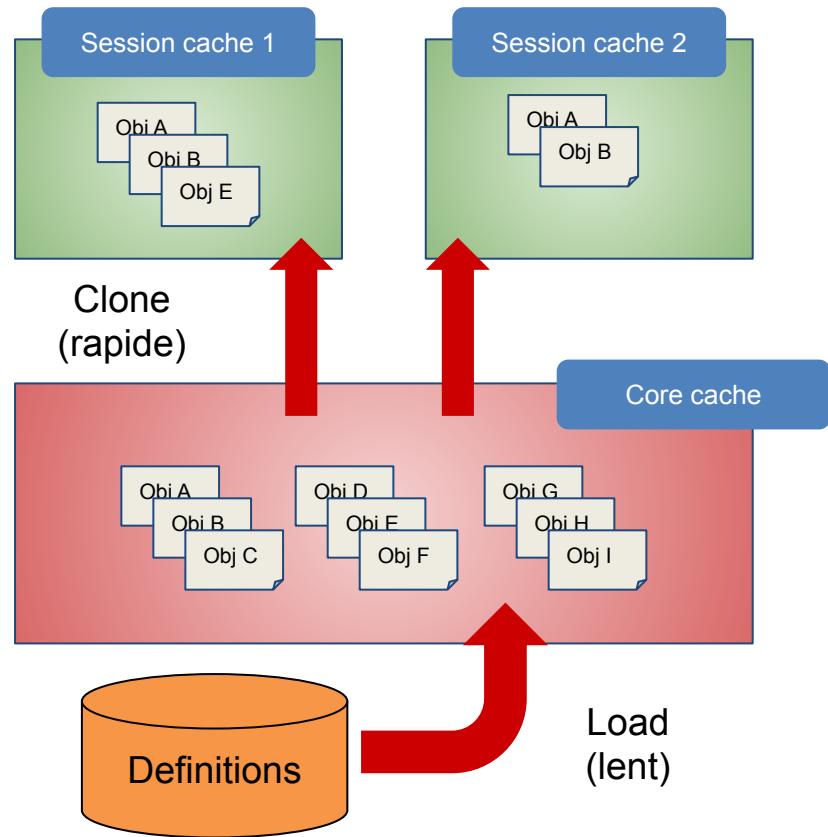


# À propos du cache

Simplicité gère ce que l'on appelle le "Core Cache" qui est une version "en mémoire" du paramétrage. **La construction d'un objet (object load) implique des mécanismes complexes:** héritages, récursivités, définitions d'attributs, nombreux appels SQL, templates, etc. Les objets étant stable en production, Simplicité ne les construit qu'une fois.

Chaque session d'utilisateur va cloner les définitions nécessaires du Core Cache dans son propre Session Cache, pour chaque instance invoquée: c'est l'action de "clone" d'objet. Chaque session conserve donc en mémoire autant de copies que nécessaire de la définition d'un objet, chacune contenant ses données.

En ce qui concerne les scripts d'objets, ils sont compilés et associés à la définition de l'objet, afin que leurs hooks soient appelés au bon moment du cycle de vie.





# Publication



- Un certain nombre d'exports par défaut sont déjà présents:
  - Excel
  - CSV
  - PDF
- Les publications sont des exports spécifiques, pouvant être générés de trois façons différentes
  - via un **template “fichier”** avec expressions
  - via un **template “simplicité”** avec expressions
  - via **code** (notamment pour Excel, PDF)
- **Exercice:** sur la commande, créer une publication faisant appel à une méthode générant un fichier excel

```
public byte[] pubExcel(PrintTemplate pt){  
    try{  
        ExcelPOITool xls = new ExcelPOITool(true);  
        ExcelPOITool.ExcelSheet sheet = xls.newSheet("Feuille 1");  
        sheet.addFullRow(0, new String[]{"A", "B", "C"}); // at line 0  
        xls.add(sheet);  
        return xls.generateToByteArray();  
    }  
    catch(Exception e){  
        AppLog.error(getClass(), "pubExcel", "Excel generation error", e,  
        getGrant());  
        return null;  
    }  
}
```



- Un lien virtuel permet d'accéder rapidement à des informations liées à un objet sans devoir naviguer dans le modèle métier.
- Un lien virtuel est une relation d'objets **sans colonne physique** et possédant un **filtre spécifique** (syntaxe SQL)
  - Créez un lien virtuel permettant d'accéder à la liste des clients d'un fournisseur.

```
t.row_id in (
    select app_client.row_id from app_client
    left join app_order on app_order.ord_client_id=app_client.row_id
    left join app_product on app_product.row_id=app_order.ord_product_id
    left join app_supplier on app_supplier.row_id=app_product.app_prd_supplier_id
    where app_supplier.row_id=[row_id]
)
```



# Arborescences

- Les arborescences sont une façon de naviguer au sein des données métier. Elles sont de deux types:
  - Construction manuelle
  - Construction Automatique
- Créez une arborescence permettant de naviguer dans les différentes commandes d'un client
  - 00** Objet métier: AppClient
  - 00-00** Objet métier: AppOrder
  - 00-00-00** Objet métier: AppProduct





# Agenda



- L'agenda permet d'afficher les données d'un objet ayant un champ “date” dans une vue calendaire.

Créez l'agenda des commandes

Toute la journée	lun. 14/1	mar. 15/1	mer. 16/1	jeu. 17/1	ven. 18/1	sam. 19/1	dim. 20/1
06							
07							
08							
09							
10							
11					10:00 - 11:00 PRD01 AST		
12							
13			13:00 - 14:00 PRD02 PANO				
14							
15							
16							
17				17:00 - 18:00 PRD04 OBE			
18							
19							
20							
21							
22							
23							

# Modèle métier

- La saisie, ainsi que la visualisation des données peut se faire via un modèle métier paramétrable.
- Créez un modèle métier permettant de créer des nouvelles commandes:
  - Créez un nouveau template de modèle
  - Ajoutez les objets du modèle
  - Ajoutez les liens du modèle
  - Habilitez votre modèle





- Un module peut être sauvegardé dans git

- Commitez votre module dans git depuis Simplicité
- Clonez votre repository git dans un IDE
- Effectuez une modification dans votre classe AppClient
- Commitez votre modification
- Vérifiez dans Simplicité que la modification a bien été prise en compte

 Application 

URI: <https://formation5.demo.simplicite.io/git/Application>

Commit message	Commit	Fermer	Supprimer	Actualiser
<a href="#">18c25e5f588e65c72af3b8c5e1f0d40777b5195f (diff to previous) (diff to current)</a> Fri Jan 18 12:06:23 CET 2019				
<a href="#">e0ca8192f5241f17c8f0e474158a5cccd82a14130 (diff to previous) (diff to current)</a> Fri Jan 18 12:03:53 CET 2019				
<a href="#">022db9c2d38c7ead455b94a8505ef8fc1d519de9 (diff to previous) (diff to current)</a> Fri Jan 18 11:57:23 CET 2019				
<a href="#">9dee77c392ce2b9686786db0a9d476320f634bla (diff to previous) (diff to current)</a> Fri Jan 18 11:56:13 CET 2019				

simplicite  
awheeler@simplicite.fr

Modification du message de Log  
[src/com/simplicite/objects/Application/AppClient.java](#)

Updated  
.classpath  
.project  
.settings/org.eclipse.core.resources.prefs  
Application.xml  
BUILD.md  
pom.xml  
sonar-project.properties  
[src/com/simplicite/objects/Application/AppClient.java](#)

designer (Designer)  
no-reply@simplicite.fr

Updated

designer (Designer)  
no-reply@simplicite.fr

Updated

designer (Designer)  
no-reply@simplicite.fr

Updated



# Objets externes



- Un objet externe Simplicité est à une page HTML accessible depuis l'UI authentifiée ou publique
- Créez un objet externe "AppExternalObject"
  - Ajoutez à votre objet une ressource HTML
  - Ajoutez à votre objet une ressource CSS
  - Habilitez votre objet
  - Ajoutez le à votre domaine

```
<div class="hi">  
    Hello World  
</div>  
  
.hi{  
    color:white;  
    background-color:blue;  
    font-size:20px;  
}
```



# Script front d'un objet métier



- Il est possible d'ajouter à un objet métier un script JavaScript

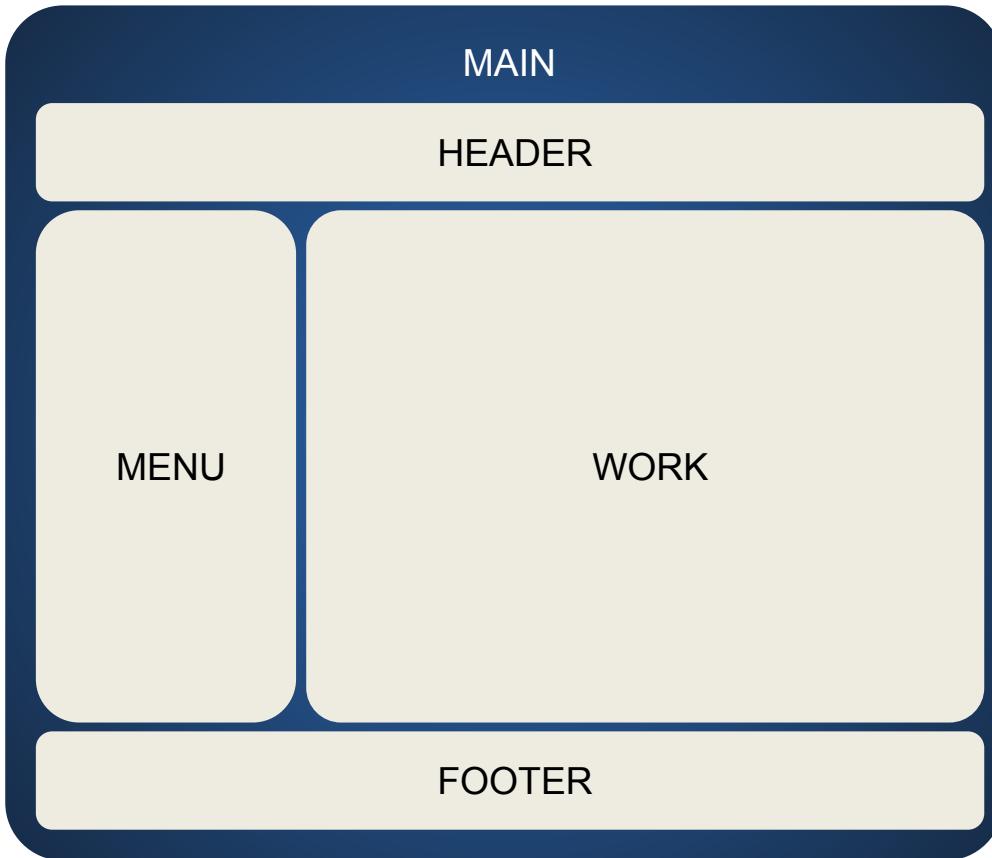
- Créez une ressource SCRIPT pour votre objet Client
- Affichez une <div> sous votre champ Nom lorsque vous le changez

```
(function(ui) {
    if (!ui) return;
    var app = ui.getAjax();
    Simplicite.UI.hooks.AppClient = function(o, cbk) {
        try {
            if (o.isMainInstance()) {
                console.log("myObject hook loading...");
                var p = o.locals.ui;
                p.form.onload = function(ctn, obj) {
                    var field = ui.getUIField(ctn, obj, "appCliLastName");
                    field.ui.on("change", function() {
                        var champ = $('#field_appCliLastName').closest(".field");
                        var warnDiv = "<div>Cette valeur vient d'être changée</div>";
                        champ.append(warnDiv);
                    });
                };
            }
            catch(e) {
                app.error("Error in Simplicite.UI.hooks.myObject: "+e.message);
            }
            finally {
                cbk && cbk();
            }
        };
    })(window.$ui);
```



## Moteur Front-End

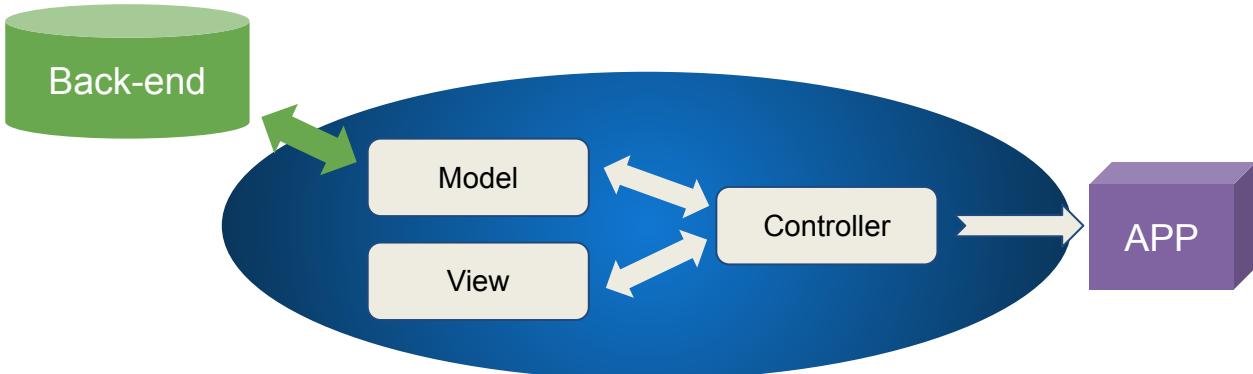
# Disposition



The disposition is responsible for serving the following front elements:

- the major HTML blocks presented on the left
- an optional “STYLES” CSS stylesheet
- an optional “SCRIPT” JS script

# Front-end engine



	CONTROLLER	MODEL	VIEW
Definition	Simplicite.UI.Engine*	Simplicite.Ajax	Simplicite.UI.View.*
Instance	\$ui	\$ui.getAjax()	\$ui.view.*

Use the definitions to override the engine behavior.  
Use the instances to work with the loaded data.

\* Simplicite.UI.Engine is the main UI controller which loads components, call the Ajax services and manages the View rendering, but there are other classes the control other specific parts (options, navigation, trays, workflows, calendars, maps).

# Examples

---

- Use object hooks
  - onChange => do sth
- Use libs (<https://www.simplicite.io/resources/thirdparty/index.md>)
- Add own libs
- Display chart in modal using main instance with filters



## Intégration

# Exercices

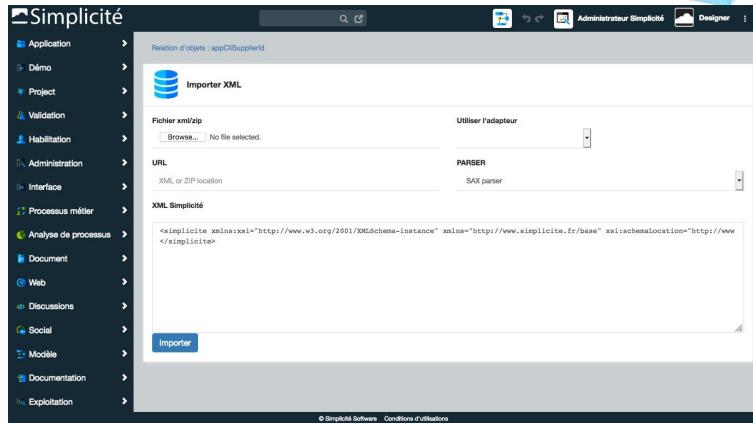
---



1. Export module (xml / zip)
2. Import module (zip)
3. Export module (git)
4. Export DATA (xml)
5. Export data (csv)
6. Import data (csv)
7. Explorer endpoints
  - o /ui
  - o /api
  - o /io
  - o /health
  - o /ext
  - o /ui/ext (ex: /ui/ext/DBAccess)

# Import de données

- Par XML ou fichier ZIP, pour les modules ou les objets



This screenshot shows the 'Importer CSV' screen. It includes a preview pane showing a table of data from 'AppClient.csv', a mapping section with 'MAPPINGS' and 'Transform' columns, and a 'Import' button at the bottom.

- Par CSV, pour les données

- Des adaptateurs peuvent être utilisés pour de l'intégration spécifique

# APIS : REST

- Endpoint /api/rest.
- HTTP
  - Curl
  - \$.ajax() jquery
  - Etc, ...

- Une interface est disponible pour tester le endpoint rest.

The screenshot shows the Simplicité API testing interface. On the left, there's a configuration panel for a service named 'designer' with a password of '\*\*\*\*\*'. The method is set to 'REST GET - application & rights'. Below this are sections for 'Options', 'Texts', and 'Texts regex'. On the right, there's a 'Response' panel showing a successful request to '/api/rest'. The response header includes:

```
Server: nginx/1.10.0
Date: Tue, 31 May 2016 15:37:11 GMT
Content-Type: application/json; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
Last-Modified: Tue, 31 May 2016 15:37:11 GMT
```

The response body is a JSON object representing the application:

```
{"application": {"name": "", "title": "Simplicité", "server": "tomcat 8", "version": "0.1", "platformversion": "3.2 M04", "encoding": "UTF-8", "database": "HSQL Database Engine 2.3.2"}, "grant": {"userId": 1, "sessionId": "82792095172227AAB02947C3A41E44B", "authToken": "#MLCTPjzUa0gyb0f6hzRsvqJPyCzeEc1o8MlkrmWPtY0h", "logIn": "designer", "lang": "FRA", "langIso": "fr", "date": "2016-05-31T10:37:11Z", "timeZone": "Europe/Paris", "rows": 20, "maxRows": 50}, "token": "", "firstName": "", "lastName": "", "designer": {"picture": {"id": "365", "name": "simplicite"}, "user": "designer", "image": "http://simplicite.com/designer/picture"}, "lang": "fr", "size": "251", "label": "User", "label2": "User", "image": "http://simplicite.com/designer/picture"}, "header": "1", "content": "7b080e04C9AA0A11E1E19aAACQAABRCAYAAABw4p0UAAAAABG8BTUEAALGPC/hnbcQAAAHw5fUAAAujgAAU1hDQgjwAAABygOrVrhol20ndhdmUAUFGFpbnQuTKVhLjUjMjTAh9HkIAAAJh0lEQVR4Xu1cDBU1xkGmaubk7MYprmpwQs6Pbkb8IrcLcogJilU830YiLoJxFBCQDxvIvhkWkYyM4TT10spnvYrdjPoW1r1FaCCQpnJmxhgtbyA78bdJn3kmnz2kvee333p9f/85i4upSKvSF/vdaw/wBuuds5hsUsbCmK9JUln1TK5kXN9b9435qgqP973n3nQ72c3D9fK9xv1Y1yDcwW1280nm3mtPgjkn+HymBovaXka5f/mh9OjOSYjanlRYEvGyGRFx+Hej1o4GnD2dzu941,6mPKFg6Uk+xZANfGBA8Ca58k+ve282z37ebnluJzbn61fPj96b7t7jQoKoKaGnV/2zdK7vKcSaCtHGDE+Tv2/IVSu4Afqf8MmQYQgJuepQh59g2D54p4hKvKz2o21TAAdC/CpcZva1LusJbumCgr8z1vtbqvFOkkekSYzR3xig1cbGxOlkKybtBS/E/le3ukgr4VpLstGz/VVYs3s4QJUih-AZ2dQhPr1ts56gSwY2RMesGyJkavdUdOcp2MttreS9QEhjlgMchmcJQg1TtgTgTmpPRkrUpJzgjTTSCEeE14l4FM/WxgREIzhmgxZzJDQpEnPxyghTjYQ3GvR4VVRChhRABohsvAZhIK0Mea1DUEKYISFICIPoNCS5YmENp3v5Nue3HjYsaSHXuSR6V0u6SgumeQ/IDLPhMqUgjm4YRlpPtkJ0WvCmrgatw4PqgVhNbk7OjaDp0YJm9Mjox9f7Q3PESjxq6ej+Cd4fQyJ5H10orJw9/AuB2MjAhB2qfM7+r4n3BUELBrn08a1ntqutnf38Rqpu294rtV1xEhCgnmtWWvzmlJUSyCamNCHY7euu7zQJlhvuk/Mu7b0lyrJ5Q5d3gvcYEB0joeOlsIEth32+ePmoQuZcI9aNLDzCylgri1-baEhaX7Ctu2g2E91uAsOnkLNEmlDzCylgri1-baEhaX7Ctu2g2E91uHPhmidvb0mVBOrvbtB/091Spnq2Zf6ue
```

# APIs : Couche Ajax

---



- Wrapper JS sur la couche JSON qui simplifie l'écriture
- Ensemble de fichiers Javascript
- Plus d'informations
  - <https://www.simplicite.io/resources/documentation/03-apis/ajax-api.md>
  - <https://www.simplicite.io/resources/4.0/jsdoc/>

# APIs : Endpoint IO

- <http://myapp/io>
- Ligne de commande : CURL
- Services
  - Import
    - XML / ZIP
    - Module
    - CSV
  - Export
    - XML / ZIP
    - Module
    - CSV
  - Clear Cache
  - Purge
    - Logs
    - Supervisions
    - Etc, ...

## Service: IMPORTXML

Login:

Password :

Optional param:

## Data:

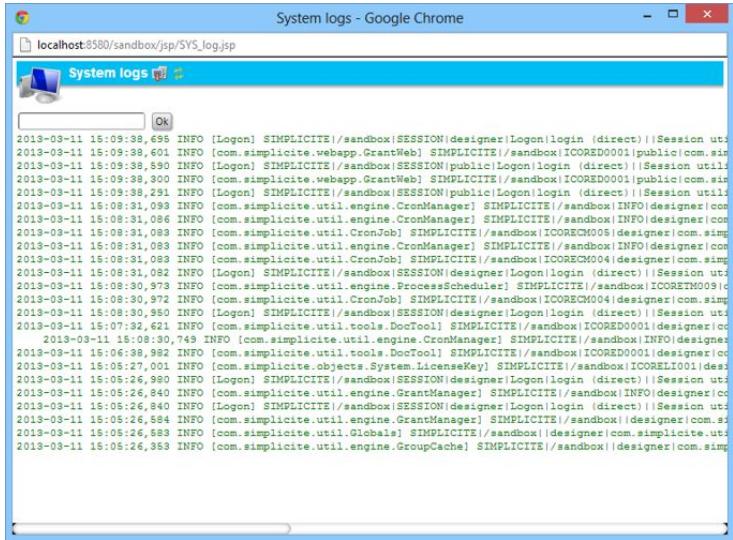
```
<?xml version="1.0" encoding="UTF-8"?>
<simplcite xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.simplcite.fr/base" xsi:schemaLocation="http://www.simplcite.fr/base
http://www.simplcite.fr/schemas/base.xsd">
</simplcite>
```



## Exploitation

# Exploitation : Logs

- Log système (serveur d'application)
  - Logs Applicatifs



Logs		<a href="#">Créer</a>		<a href="#">Créer en liste</a>	<a href="#">Supprimer les logs</a>	<a href="#">Purge partielle des logs</a>	<a href="#">Voir les LOG serveur</a>			Synthèse	
Date ^	Utilisateur	Plateforme	Code Événement	Niveau Événement	Libellé Événement	Sujet		Id interne	Méthode	Trace1	
Depuis											
Jusqu'à											
16/01/2019 18:10:03	system	https://demo.simplicite.io:15123	PLATFORM		Platform message	com.simplicite.util.engine.Platform			addNode	Registered node = https://dem...	
16/01/2019 18:15:00	system	https://demo.simplicite.io:15123	SESSION		Événement de login/logout	Logon			login (direct)	Session user=system/job_Impo...	
16/01/2019 18:15:00	system	https://demo.simplicite.io:15123	SESSION		Événement de login/logout	Logon			login (direct)	Session user=system/job_dead	
16/01/2019 18:20:01	system	https://demo.simplicite.io:15123	SESSION		Événement de login/logout	Logon			login (direct)	Session user=system/job_Obje...	
16/01/2019 18:33:00	system	https://demo.simplicite.io:15123	SESSION		Événement de login/logout	Logon			login (direct)	Session user=system/job_Obje...	
16/01/2019 19:00:01	system	https://demo.simplicite.io:15123	SESSION		Événement de login/logout	Logon			login (direct)	Session user=system/job_dead	

# Exploitation : Logs (2/2)

- Toutes les logs sont disponibles dans la console du navigateur si les websocket sont activées



```
Developer Tools - http://demo.apps.simplicite.io/jsp/index.jsp
Audits Elements Console Sources Network Timeline Profiles Resources Security
✖ ✖ top ▼  Preserve log

Event websocket URL = ws://demo.apps.simplicite.io/jsp/events
Events websocket opened

SESSION|designer|Logon|login (direct)|Session utilisateur=designer/AC2353A83F5C1279A8AB98B506F4CD61 time=0
> |
```

# Exploitation : Supervision des imports

- Tous les fichiers importés (XML, CSV, ...) sont supervisés.

Supervision imports														
Actions	Statut	Utilisateur	Origine	Créé le	Date d'effet	Importé le	Durée (sec)	Source	Fichier	Log	Erreur / Rejets	Nom Adaptateur	Type traitement Adaptateur	
												Import CSV	Import XML	Imports différés
<input type="checkbox"/>	<input checked="" type="checkbox"/>	system	Import module: Markdown	22/01/2019 04:00:24	22/01/2019 04:00:25	22/01/2019 04:00:25	22,372	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				
<input type="checkbox"/>	<input checked="" type="checkbox"/>	designer	I/O XML import https://www.simplicité.io/resources/modules/demo-app-4.0.xml	21/01/2019 17:26:36	21/01/2019 17:26:36	21/01/2019 17:26:36	6,456	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				
<input type="checkbox"/>	<input checked="" type="checkbox"/>	designer	I/O XML import	21/01/2019 17:25:48	21/01/2019 17:25:48	21/01/2019 17:25:48	0,981	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				
<input type="checkbox"/>	<input checked="" type="checkbox"/>	system	Import module: Markdown	21/01/2019 04:00:21	21/01/2019 04:00:21	21/01/2019 04:00:21	24,963	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				
<input type="checkbox"/>	<input checked="" type="checkbox"/>	system	Import module: Markdown	20/01/2019 04:00:22	20/01/2019 04:00:23	20/01/2019 04:00:23	25,761	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				
<input type="checkbox"/>	<input checked="" type="checkbox"/>	system	Import module: Markdown	19/01/2019 04:00:29	19/01/2019 04:00:29	19/01/2019 04:00:29	27,620	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				
<input type="checkbox"/>	<input checked="" type="checkbox"/>	system	Import module: Markdown	18/01/2019 04:00:25	18/01/2019 04:00:25	18/01/2019 04:00:25	26,878	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				
<input type="checkbox"/>	<input checked="" type="checkbox"/>	designer	UNDO REDO PATCH XML	17/01/2019	17/01/2019	17/01/2019	0.572	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				

# Exploitation : Tâches planifiées

- Tâches planifiées avec la syntaxe CRON.
  - Lancer la tâche
  - Planifier la tâche
  - Résumé des exécutions

The screenshot shows a web-based application titled "CronTable". The main area displays a table of scheduled tasks. The columns include: Name, Function, Object, Action, Method, Cron expression, Enabled, and Run as Login. The table lists various tasks such as "manageDeadlineActivity", "manageDeadlineProcess", "manageDeadlineState", "manageDeadlockActivity", "HealthCheck", "Version", "ImportXML", "ManageImportXML", "LicenseReminder", "LogCacheMemory", "LogDiskMemory", "LogDocMemory", "LogMemory", "LogSession", "LogSystem", "ObjectDynGC", and "ObjectFullGC". The "Cron expression" column shows standard cron expressions like "0 0/1 \* \* ?" or "0 0/2 \* \* ?". The "Enabled" column contains checkboxes, all of which are checked. The "Run as Login" column contains checkboxes, all of which are checked.

Name	Function	Object	Action	Method	Cron expression	Enabled	Run as Login
deadlineActivity	deadlineActivity	BPMActivity	deadlineActivity	manageDeadlineActivity	0 0/1 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
deadlineProcess	deadlineProcess	BPMProcess	deadlineProcess	manageDeadlineProcess	0 0/1 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
deadlineState	deadlineState	BPMState	deadlineState	manageDeadlineState	0 0/1 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
deadlockActivity	deadlockActivity	BPMActivity	deadlockActivity	manageDeadlockActivity	0 0/15 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
HealthCheck	Version	SystemParam	getVersion		0 0/1 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ImportXML	ManageImportXML	XMLSupervisor	manageImport		0 0/5 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ImportXML	ManageImportXML	XMLSupervisor	manageImport		0 0/5 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
LicenseReminder	LicenseReminder	LicenseKey	LicenseReminder		0 15 10 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
LogCacheMemory	LogCacheMemory	AppLoggerMemory	LogCacheMemory	logMemoryCache	0 0/2 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
LogDiskMemory	LogDiskMemory	AppLoggerMemory	LogDiskMemory	logMemoryDisk	0 0/2 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
LogDocMemory	LogDocMemory	AppLoggerMemory	LogDocMemory	logMemoryDoc	0 0/2 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
LogMemory	LogMemory	AppLoggerMemory	LogMemory		0 0/2 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
LogSession	LogSession	AppLoggerMemory	LogSession	logSession	0 0/2 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
LogSystem	LogSystem	AppLoggerMemory	LogSystem	logSystem	0 0/2 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ObjectDynGC	CronTable-DynGC	CronTable	ObjectDynGC	objectDynGC	0 5/15 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ObjectFullGC	CronTable-FullGC	CronTable	ObjectFullGC	objectFullGC	0 3/30 * * ?	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

The screenshot shows a dialog box titled "Update CronTable : LogCacheMemory" and a table below it. The dialog box contains fields for "Name" (LogCacheMemory), "Function" (LogCacheMemory), "Object" (AppLoggerMemory), "Action" (LogCacheMemory), and "Method" (logMemoryCache). The "Cron expression" field is set to "0 0/2 \* \* ?". The "Enabled" checkbox is checked. The "Run as Login" checkbox is checked. The "Jobs depth" is set to "-3". The "Description" is "Log the Cache memory". The "Module Name" is "WorkflowUser". At the bottom, there are "Save", "Save & Close", and "Close" buttons, along with a "Force immediate execution of task" button. Below the dialog is a table titled "Asynchronous job" showing three entries. The columns include: UID, Status, Start date, End date, Object instance, and Action. The entries are: 5822-14158605039271-1207374051 (Terminated, 11/13/2014 08:50:39, 11/13/2014 08:50:39, job\_AppLoggerMemory, LogCacheMemory); 5814-1415864919240-66335564 (Terminated, 11/13/2014 08:48:39, 11/13/2014 08:48:39, job\_AppLoggerMemory, LogCacheMemory); and 5807-1415864793224-1060031834 (Terminated, 11/13/2014 08:46:39, 11/13/2014 08:46:39, job\_AppLoggerMemory, LogCacheMemory).

UID	Status	Start date	End date	Object instance	Action
5822-14158605039271-1207374051	Terminated	11/13/2014 08:50:39	11/13/2014 08:50:39	job_AppLoggerMemory	LogCacheMemory
5814-1415864919240-66335564	Terminated	11/13/2014 08:48:39	11/13/2014 08:48:39	job_AppLoggerMemory	LogCacheMemory
5807-1415864793224-1060031834	Terminated	11/13/2014 08:46:39	11/13/2014 08:46:39	job_AppLoggerMemory	LogCacheMemory



# Simplicité

BRING YOUR OWN BUSINESS



[Linkedin](#)



[Twitter](#)



[Facebook](#)

[www.simplicite.fr](http://www.simplicite.fr)

[www.simplicitesoftware.com](http://www.simplicitesoftware.com)



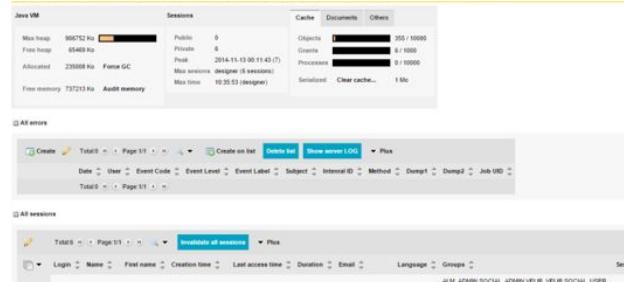
38 rue de Trévise  
75009 Paris



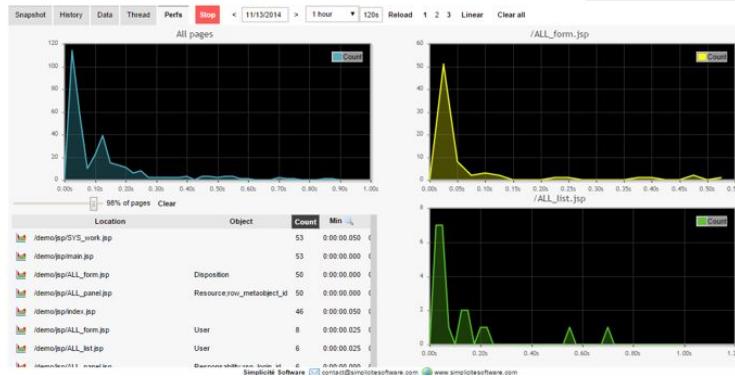
# Exploitation : Monitoring



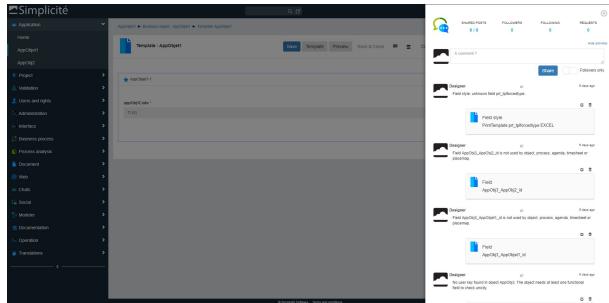
- Vide le cache global, par session, ...



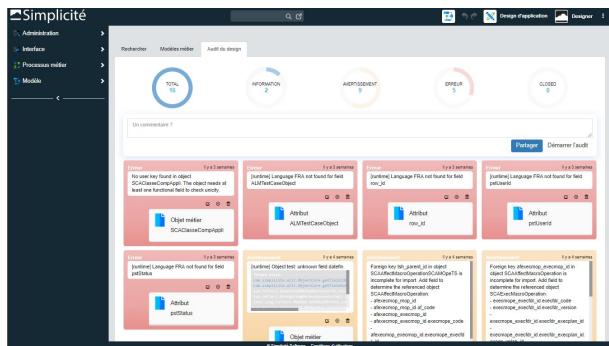
- Graphiques
  - JVM
  - Mémoires
  - Requêtes SQL



# Plateforme collaborative



- Pour les utilisateurs métier ou les exploitants sur la base de messages partagés sur des données ou sur un thème de discussion générale.



- Pour concepteurs d'applications au travers de l'audit du design (aide à la conception).

# Disposition du site

Disposition > Disposition : responsive

Disposition : responsive

Enregistrer Enregistrer & Fermer Vider le cache Editer le code Fermer

Code \* Module \*

responsive UI

Etendre ▾

Vue

Paramètre de disposition

Application

Ressource 15

Utilisation code partagé

Implémentation

Ressource

Créer

Créer en liste

Editer la liste

Type	Langue	Code	Mise en cache ?	Fichier	Image	Nom Module
Style CSS	*	STYLES	Non			UI
HTML	Anglais	FOOTER	Non			UI
HTML	*	FOOTER	Non			UI
HTML	*	HEADER	Non			UI
HTML	*	MAIN	Non			UI
HTML	*	MENU	Non			UI
HTML	*	WORK	Non			UI

© Simplicité Software Conditions d'utilisation

Les ressources de la disposition responsive